(12) DEMANDE INTERNATIONALE PUBLIÉE EN VERTU DU TRAITÉ DE COOPÉRATION EN MATIÈRE DE BREVETS (PCT)

(19) Organisation Mondiale de la Propriété Intellectuelle
Bureau international

(43) Date de la publication internationale
9 septembre 2005 (09.09.2005)

PCT

(10) Numéro de publication internationale
WO 2005/083889 A1

(51) Classification internationale des brevets[7] : H03M 7/30

(21) Numéro de la demande internationale :
PCT/FR2004/000219

(22) Date de dépôt international :
30 janvier 2004 (30.01.2004)

(25) Langue de dépôt : français

(26) Langue de publication : français

(71) Déposant *(pour tous les États désignés sauf US)* : **FRANCE TELECOM** [FR/FR]; 6, place d'Alleray, F-75015 Paris (FR).

(72) Inventeurs; et

(75) Inventeurs/Déposants *(pour US seulement)* : **LAMBLIN, Claude** [FR/FR]; Résidence Pont Couennec, F-22700 Perros Guirec (FR). **VIRETTE, David** [FR/FR]; 25, Avenue Ernest Renan, F-22300 Lannion (FR). **KOVESI, Balazs** [HU/FR]; 12 résidence Corlay, F-22300 Lannion (FR). **MASSALOUX, Dominique** [FR/FR]; 53, rue du Pré de Saint-Maur, F-22700 Perros-Guirec (FR).

(74) Mandataires : **LOUISET Raphaël** etc.; Cabinet Plasseraud, 65/67, Rue de la Victore, F-75440 Paris Cedex 9 (FR).

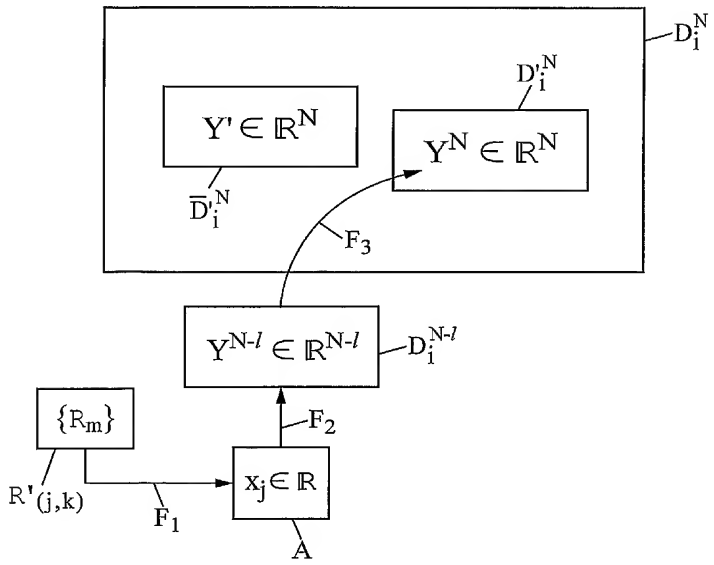
(81) États désignés *(sauf indication contraire, pour tout titre de protection nationale disponible)* : AE, AG, AL, AM, AT, AU, AZ, BA, BB, BG, BR, BW, BY, BZ, CA, CH, CN, CO, CR, CU, CZ, DE, DK, DM, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, HR, HU, ID, IL, IN, IS, JP, KE, KG, KP, KR, KZ, LC, LK, LR, LS, LT, LU, LV, MA, MD, MG, MK, MN, MW, MX, MZ, NA, NI, NO, NZ, OM, PG, PH, PL, PT, RO, RU, SC, SD, SE, SG, SK, SL, SY, TJ, TM, TN, TR, TT, TZ, UA, UG, US, UZ, VC, VN, YU, ZA, ZM, ZW.

(84) États désignés *(sauf indication contraire, pour tout titre de protection régionale disponible)* : ARIPO (BW, GH, GM, KE, LS, MW, MZ, SD, SL, SZ, TZ, UG, ZM, ZW), eurasien

[Suite sur la page suivante]

(54) Title: DIMENSIONAL VECTOR AND VARIABLE RESOLUTION QUANTISATION

(54) Titre : QUANTIFICATION VECTORIELLE EN DIMENSION ET RESOLUTION VARIABLES



(57) Abstract: The invention relates to compression coding and/ or decoding of digital signals, in particular by vector variable-rate quantisation defining a variable resolution. For this purpose an impulsion dictionary comprises: for a given dimension, increasing resolution dictionaries imbricated into each other and, for a given dimension, a union of: a totality (D'_i<SP>N</SP>) of code-vectors produced, by inserting elements taken in a final set (A) into smaller dimension code-vectors according to a final set of predetermined insertion rules (F1) and a second totality of code-vectors (Y') which are not obtainable by insertion into the smaller dimension code vectors according to said set of the insertion rules.

[Suite sur la page suivante]

WO 2005/083889 A1

(AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), européen (AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, HU, IE, IT, LU, MC, NL, PT, RO, SE, SI, SK, TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, ML, MR, NE, SN, TD, TG).

Déclaration en vertu de la règle 4.17 :

— *relative à la qualité d'inventeur (règle 4.17.iv)) pour US seulement*

Publiée :

— *avec rapport de recherche internationale*

En ce qui concerne les codes à deux lettres et autres abréviations, se référer aux "Notes explicatives relatives aux codes et abréviations" figurant au début de chaque numéro ordinaire de la Gazette du PCT.

(57) Abrégé : La présente invention concerne le codage et/ou décodage en compression de signaux numériques, en particulier par quantification vectorielle à débit variable définissant une résolution variable. Elle vise à cet effet un dictionnaire à impulsions comportant d'une part, pour une dimension donnée, des dictionnaires de résolution croissante imbriqués les uns dans les autres, et, d'autre part, pour une dimension donnée, une union d'un ensemble (D'^{N}_{i}) de vecteurs-codes construits en insérant, dans des vecteurs-codes de dimension inférieure, des éléments pris dans un ensemble fini (A) selon un jeu fini de règles d'insertion prédéterminées (F^{l}), et d'un deuxième ensemble constitué de vecteurs-codes (Y') ne pouvant être obtenus par insertion dans des vecteurs-codes de dimension inférieure selon ce jeu de règles d'insertion.

QUANTIFICATION VECTORIELLE EN DIMENSION ET RESOLUTION VARIABLES

La présente invention concerne le codage et/ou décodage en compression de signaux numériques tels que les signaux audio, vidéo, et plus généralement les signaux multimédia pour leur stockage et/ou leur transmission.

Une solution très répandue en compression des signaux numériques est la quantification vectorielle. Une première incitation à utiliser la quantification vectorielle peut être trouvée dans la théorie du codage en blocs développée par Shannon selon laquelle une meilleure performance peut être atteinte en augmentant la dimension des vecteurs à coder. La quantification vectorielle consiste à représenter un vecteur d'entrée par un vecteur de même dimension choisi dans un ensemble fini. Ainsi, prévoir un quantificateur à M niveaux (ou vecteurs-codes) revient à créer une application non bijective de l'ensemble des vecteurs d'entrée (généralement l'espace réel euclidien à n dimensions R^n, ou encore un sous-ensemble de R^n) dans un sous-ensemble fini Y de R^n. Le sous-ensemble Y comporte alors M éléments distincts :

$$Y = \{y_1, y_2, \ldots y_M\}.$$

Y est appelé alphabet de reproduction, ou encore dictionnaire, ou encore répertoire. Les éléments de Y sont dits "*vecteurs-codes*", "*mots de code*", "*points de sortie*", ou encore "*représentants*".

Le débit par dimension (r) du quantificateur (ou encore sa

"*résolution*") est défini par :

$$r = \frac{1}{n} \log_2 M$$

En quantification vectorielle, un bloc de n échantillons est traité comme un vecteur de dimension n. Le vecteur est codé en choisissant un vecteur-code, dans un dictionnaire de M vecteurs-codes, celui qui lui "*ressemble*" le plus. En général, une recherche exhaustive est faite parmi tous les éléments du dictionnaire pour sélectionner l'élément du dictionnaire qui minimise une mesure de distance entre lui et le vecteur d'entrée.

Selon la théorie du codage de source, quand la dimension devient trop grande, la performance de la quantification vectorielle approche une limite dite "*borne de débit-distorsion de la source*". Outre la dimensionnalité de l'espace, la quantification vectorielle peut aussi exploiter les propriétés de la source à coder, par exemple des dépendances non-linéaires et/ou linéaires, ou encore la forme de la distribution de probabilité. En général, les dictionnaires de quantificateurs vectoriels sont conçus à partir de méthodes statistiques telles que l'algorithme de Lloyd généralisé (noté GLA pour "*Generalized Lloyd Algorithm*"). Cet algorithme, bien connu, est basé sur les conditions nécessaires d'optimalité d'une quantification vectorielle. A partir d'une séquence d'entraînement représentative de la source à coder et d'un dictionnaire initial, le dictionnaire est construit de façon itérative. Chaque itération comprend deux étapes :

- la construction des régions de quantification par quantification de la séquence d'entraînement selon la règle du plus proche voisin, et
- l'amélioration du dictionnaire en remplaçant les anciens vecteurs-codes par les centroïdes des régions (selon la règle des centroïdes).

Pour éviter la convergence vers un minimum local de cet algorithme itératif déterministe, des variantes dites de "*relaxation stochastique*" (notées SKA pour "*Stochastic K-means algorithm*") inspirées de la technique du recuit simulé ont été proposées en introduisant une part d'aléatoire dans l'étape de construction des centroïdes et/ou dans celle de construction des classes. Les quantificateurs vectoriels statistiques ainsi obtenus ne possèdent aucune structure, ce qui rend leur exploration coûteuse en calculs et gourmande en mémoire. En effet, la complexité tant du codage que du stockage, est proportionnelle à $n.2^{nr}$. Cette croissance exponentielle en fonction de la dimension des vecteurs et du débit limite l'emploi des quantificateurs vectoriels non structurés à de faibles dimensions et/ou de bas débits pour pouvoir les implanter en temps réel.

La quantification scalaire, qui quantifie les échantillons de façon individuelle, n'est pas aussi efficace que la quantification vectorielle car elle ne peut exploiter que la forme de la distribution de probabilité de la source et la dépendance linéaire. Toutefois, la quantification scalaire est moins coûteuse en calculs et en mémoire que la quantification vectorielle. De plus, la quantification

scalaire associée à un codage entropique peut atteindre de bonnes performances même à des résolutions modérées.

Pour s'affranchir des contraintes de taille et de dimension, plusieurs variantes de la quantification vectorielle de base furent étudiées, elles tentent de remédier à l'absence de structure du dictionnaire et parviennent ainsi à réduire la complexité au détriment de la qualité. Cependant, le compromis performance/complexité est amélioré, ce qui permet d'accroître la plage des résolutions et/ou des dimensions sur laquelle la quantification vectorielle peut être appliquée efficacement en coût de calculs ou de mémoire.

De nombreux schémas de quantificateurs vectoriels structurés ont été proposés dans la littérature. Les principaux sont les suivants :

- le quantificateur vectoriel en arbre qui impose au dictionnaire une structure hiérarchique en arbre : la procédure de recherche est simplifiée mais le quantificateur nécessite plus de mémoire de stockage,
- le quantificateur vectoriel à étages multiples qui met en cascade des quantificateurs vectoriels de niveaux moindres : les dictionnaires sont de tailles réduites et il en va de même pour ce qui concerne le temps de calcul et le coût en mémoire,
- le quantificateur vectoriel dit "*produit cartésien*" de N quantificateurs vectoriels classiques de tailles et de dimensions plus petites : on décompose le vecteur d'entrée en N sous-vecteurs, chaque sous-vecteur étant quantifié indépendamment des autres,

- le quantificateur vectoriel *"gain/orientation"* constitue un cas particulier du quantificateur vectoriel *"produit cartésien"* : on prévoit deux quantificateurs, l'un scalaire et l'autre vectoriel, qui codent séparément, de façon indépendante ou non, le gain (ou la norme) du vecteur et son orientation (en considérant le vecteur d'entrée normalisé). Ce type de quantification vectorielle est aussi appelé quantification vectorielle "*sphérique*" ou quantification vectorielle "*polaire*",
- le quantificateur vectoriel *"code à permutation"*, dont les vecteurs-codes sont obtenus par permutations des composantes d'un vecteur-leader et sa généralisation à la composée (ou l'union) de codes à permutation.

Les techniques décrites ci-dessus relèvent toutes d'une approche statistique.

Une autre approche radicalement différente a aussi été proposée. Il s'agit de la quantification vectorielle algébrique, qui utilise des dictionnaires fortement structurés, issus de réseaux réguliers de points ou des codes correcteurs d'erreur. Grâce aux propriétés algébriques de leurs dictionnaires, les quantificateurs vectoriels algébriques sont simples à mettre en oeuvre et n'ont pas à être stockés en mémoire. L'exploitation de la structure régulière de ces dictionnaires permet en effet le développement d'algorithmes de recherche optimaux et rapides et de mécanismes pour associer en particulier un indice (ou "*index*") à un vecteur-code correspondant (par exemple par une formule). Les quantificateurs vectoriels

algébriques sont moins complexes à mettre en oeuvre et nécessitent moins de mémoire. Toutefois, ils ne sont optimaux que pour une distribution uniforme de la source (soit dans l'espace, soit à la surface d'une hyper-sphère). S'agissant d'une généralisation du quantificateur scalaire uniforme, le quantificateur vectoriel algébrique est plus difficile à ajuster à la distribution de la source par la technique dite du "*companding*". On rappelle aussi que l'indexation (ou numérotation) des vecteurs-codes et l'opération inverse (décodage) nécessitent plus de calculs que dans le cas des quantificateurs vectoriels statistiques, pour lesquels ces opérations sont effectuées par de simples lectures de table.

On présente ci-après certains aspects d'une quantification à dimension variable et les problèmes rencontrés.

On indique d'abord que la quantification vectorielle est une technique bien connue et efficace pour coder des blocs d'échantillons de longueur fixe. Cependant, dans de nombreuses applications de compression du signal numérique, le signal à coder est modélisé par une séquence de paramètres de longueur variable. Une compression efficace de ces vecteurs de dimension variable est cruciale pour la conception de beaucoup de codeurs multimédia tels que les codeurs de parole ou audio (codeur "*MBE*", codeur harmonique, codeur sinusoïdal, codeur par transformée, codeur par interpolation de formes d'onde prototypes).

Dans les codeurs sinusoïdaux, le nombre de sinusoïdes

extraites dépend du nombre de pics sinusoïdaux détectés dans le signal, nombre qui varie au cours du temps en fonction de la nature du signal audio.

En outre, de nombreuses techniques de compression de la parole exploitent la périodicité à long terme du signal. C'est le cas des codeurs harmoniques où les composantes spectrales d'un ensemble de fréquences, qui sont les harmoniques de la période fondamentale du locuteur, sont codées. Le nombre de pics harmoniques spectraux étant inversement proportionnel à la fréquence fondamentale, comme cette période de fondamental varie selon le locuteur (typiquement, les enfants ayant une fréquence de vibration des cordes vocales plus haute que les hommes) et au cours du temps, le nombre de composantes à quantifier change aussi au cours du temps de trame à trame.

C'est aussi le cas des codeurs PWI (pour "*Prototype Waveform Interpolation*") où les formes d'onde prototype sont extraites sur des segments de longueur égale à la période du pitch, donc aussi variables temporellement. Dans les codeurs PWI, la quantification de ces formes d'onde de longueur variable est effectuée en codant séparément le gain (ou "*RMS*" pour "*Root-Mean-Square*") et la forme d'onde normalisée qui est elle-même décomposée en deux formes d'ondes de même longueur variable : la forme d'onde REW ("*Rapidly Evolving Waveform*") et la forme d'onde SEW ("*Slowly Evolving Waveform*"). Pour une trame de longueur fixe, le nombre de prototypes est variable, donc le nombre de gains, de REW et SEW est lui aussi variable, ainsi que la dimension des formes d'ondes REW et SEW.

Dans d'autres types de codeurs, tels que les codeurs audio par transformée, le nombre de coefficients de transformée obtenus sur des longueurs de trame de longueur fixe est imposé mais il est usuel de regrouper ces coefficients en bandes de fréquence pour les quantifier. Classiquement, cette découpe est effectuée en bandes de largeurs inégales pour exploiter les propriétés psychoacoustiques de l'audition humaine en suivant les bandes critiques de l'oreille. La plage de variation de la dimension de ces vecteurs de coefficients de transformée varie typiquement de 3 (pour les bandes de plus basses fréquences) à 15 (pour les bandes de hautes fréquences), dans un codeur en bande élargie (50Hz-7000Hz), et même jusqu'à 24 dans un codeur en bande FM (couvrant la gamme audible 20Hz-16000Hz).

Théoriquement, un quantificateur vectoriel optimal de dimension variable emploierait un ensemble de dictionnaires de dimension fixe, un pour chaque dimension possible du vecteur d'entrée. Par exemple, dans les codeurs harmoniques, pour une période de pitch de 60 à 450 Hz, le nombre de pics harmoniques en bande téléphonique variant de 7 pour les voix aiguës (enfants) à 52 pour les voix graves (hommes), il faudrait construire, mettre en mémoire et en oeuvre 46 (46=52-7) quantificateurs vectoriels. La conception de chaque dictionnaire nécessite une séquence d'apprentissage suffisamment longue pour représenter correctement les statistiques des vecteurs d'entrée. De plus, le stockage de tous les dictionnaires se révèle impraticable ou très coûteux en mémoire. On voit

donc que dans le cas de dimension variable, il est difficile de tirer parti des avantages de la quantification vectorielle en respectant des contraintes de stockage mémoire et aussi de séquences d'entraînement.

On présente ci-après certains aspects d'une quantification à résolution variable et les problèmes rencontrés.

On précise d'abord que la variabilité du signal d'entrée ne se traduit pas seulement par la variation du nombre de paramètres à coder mais aussi par la variation de la quantité d'informations binaires à transmettre pour une qualité donnée. Par exemple en parole, les attaques ("*onset*"), les sons voisés et les sons non voisés ne nécessitent pas le même débit pour une même qualité. Les attaques peu prédictibles nécessitent un débit plus élevé que les sons voisés plus stables et dont la stationnarité peut être mise à profit par des "*prédicteurs*" qui permettent de réduire le débit. Enfin, les sons non voisés ne nécessitent pas une grande précision de codage et donc requièrent peu de débit.

Pour exploiter la variation temporelle des caractéristiques des signaux multimédias tels que la voix ou la vidéo, il est judicieux de concevoir des codeurs à débit variable. Ces codeurs à débit variable sont particulièrement adaptés aux communications sur réseaux, par paquets, tels que l'Internet, l'ATM, ou autres.

En effet, la commutation par paquets permet de manipuler et traiter de façon plus flexible les bits d'information

et donc d'augmenter la capacité du canal en réduisant le débit moyen. L'utilisation de codeurs à débit variable est aussi un moyen efficace de lutter contre la congestion du système et/ou de s'adapter à la diversité des conditions d'accès.

Dans les communications multimédias, les quantificateurs à débit variable permettent aussi d'optimiser la répartition du débit entre :

- les codages source et canal : comme dans le concept de l'AMR ("*Adaptive Multi Rate*"), le débit peut être commuté à chaque trame de 20 ms pour être adapté dynamiquement aux conditions d'erreurs canal et de trafic. La qualité globale de la parole est ainsi améliorée en assurant une bonne protection contre les erreurs, tout en réduisant le débit pour le codage de la source si le canal se dégrade;
- les différents types de signaux média (tels que la voix et la vidéo dans les applications de visioconférence);
- les différents paramètres d'un même signal : dans les codeurs audio par transformée, par exemple, il est usuel de répartir dynamiquement les bits entre l'enveloppe spectrale et les différentes bandes de coefficients. Souvent, un codage entropique de l'enveloppe est d'abord effectué et a pour objectif d'exploiter la distribution non uniforme des mots de code en assignant des codes de longueur variable aux mots de code, les plus probables ayant une longueur plus courte que les moins probables, ce qui conduit à minimiser la longueur moyenne des mots de code. De plus, pour exploiter les propriétés psychoacoustiques

de l'oreille humaine, le débit restant (variable) est alloué dynamiquement aux bandes fréquentielles des coefficients en fonction de leur importance perceptuelle.

Les nouvelles applications de codage multimédia (telles que l'audio et la vidéo) nécessitent des quantifications hautement flexibles tant en dimension qu'en débit. La gamme de débits devant en plus permettre d'atteindre une haute qualité, ces quantificateurs multidimensionnels et multi-résolutions doivent viser des hautes résolutions. La barrière de complexité posée par ces quantificateurs vectoriels reste, en soi, une performance à atteindre, malgré l'augmentation des puissances de traitement et des capacités mémoire des nouvelles technologies.

Comme on le verra ci-après, la plupart des techniques de codage de source proposées visent soit à résoudre les problèmes liés à une dimension variable, soit les problèmes liés à une résolution variable. Peu de techniques aujourd'hui proposées permettent de résoudre conjointement ces deux problèmes.

Pour ce qui concerne la quantification vectorielle à dimension variable, connue, la variabilité de la dimension des paramètres à coder constitue en soi un obstacle à l'utilisation de la quantification vectorielle. Ainsi, les premières versions du codeur par transformée emploient des quantificateurs scalaires de Lloyd-Max. Un codeur de ce type, dit "*TDAC*", qu'a développé la Demanderesse, est décrit notamment dans :

- *"High Quality Audio Transform Coding at 64 kbit/s"*, de Y.Mahieux, J.P.Petit, dans IEEE Trans. Commun, Vol. 42, No 11, pp. 3010-3019, Novembre 1994.

D'autres solutions ont été proposées pour résoudre ce problème de quantification vectorielle de dimension variable. Le codeur "*IMBE*" utilise un schéma de codage compliqué avec des allocations binaires variables et une quantification hybride scalaire/vectorielle.

Une approche très couramment utilisée pour quantifier des vecteurs de dimension variable consiste à pré-traiter le vecteur de dimension variable pour le convertir en un autre vecteur de dimension fixe avant la quantification. Il existe plusieurs variantes de cette technique de quantification vectorielle associée à une conversion de dimension (ce type de quantification vectorielle étant noté DCVQ pour "*Dimension Conversion Vector Quantization*").

Parmi les différentes procédures de conversion de dimension proposées, on peut citer notamment : la troncature, le sous-échantillonnage, l'interpolation, le *"length warping"*.

Pour les codeurs de parole sinusoïdaux ou MBE, il a été proposé d'approximer les coefficients spectraux par un modèle tout-pôle d'ordre fixe puis d'effectuer une quantification vectorielle de dimension fixe des paramètres du modèle. Une autre technique de quantification vectorielle par transformée matricielle non

carrée résout le problème de la quantification vectorielle de dimension variable L en combinant une quantification vectorielle de dimension fixe K (K<L) avec une transformée linéaire matricielle non carrée (LxK).

On note aussi un autre type de quantification vectorielle associée à une conversion de dimension qui utilise toujours un quantificateur vectoriel de dimension fixe K mais la conversion de dimension est appliquée aux vecteurs-codes pour obtenir des vecteurs-codes ayant la même dimension que le vecteur d'entrée.

L'inconvénient de la quantification vectorielle associée à une conversion de dimension est que la distorsion totale a deux composantes: l'une due à la quantification, l'autre à la conversion de dimension. Pour éviter cette distorsion due à la conversion de dimension, une autre approche de la quantification vectorielle de dimension variable consiste à considérer chaque vecteur d'entrée de dimension variable L comme formé par un sous-ensemble de composantes d'un vecteur "*sous-jacent*" de dimension K (L<K) et à ne concevoir et n'utiliser qu'un seul dictionnaire "*universel*" de dimension fixe K qui couvre cependant toute la plage des dimensions des vecteurs d'entrée, la correspondance entre le vecteur d'entrée étant effectuée par un sélecteur. Cependant, ce dictionnaire "*universel*" englobant tous les autres dictionnaires de dimensions inférieures ne paraît pas optimal pour les dimensions plus faibles. En particulier, la résolution maximale r_{max} par dimension est limitée par la contrainte de stockage et par le débit par vecteur de paramètres. Pour un dictionnaire

de taille $2^{Kr_{max}}$, la quantité de mémoire requise pour stocker ce dictionnaire est $K2^{Kr_{max}}$ valeurs et son débit par vecteur de paramètres est de Kr_{max}. Ainsi, pour une même taille de dictionnaire (et donc un même débit par vecteur de paramètres et par trame), un vecteur de dimension L (L<K) pourrait avoir une résolution (ou un débit par dimension) K/L fois plus grande, et ce, pour un volume d'informations à stocker K/L fois plus petit.

Pour ce qui concerne la quantification vectorielle à résolution variable, connue, une solution simple consiste à, comme pour le cas de la quantification vectorielle à dimension variable, utiliser une quantification scalaire, comme par exemple dans les premières versions de codeur par transformée TDAC.

Cependant, l'utilisation d'une résolution entière par échantillon entraîne une granularité de résolution grossière par bande de coefficients qui nuit à l'efficacité de la procédure d'allocation binaire dynamique. On a alors proposé d'utiliser des quantificateurs scalaires à nombre entier impair de niveaux de reconstruction, en combinaison avec une procédure de mise en train binaire conjointe des indices codés. La granularité plus fine de la résolution apportée, plus propice à la procédure d'allocation binaire, a permis d'améliorer la qualité, au prix d'une complexité de l'algorithme de combinaison des indices, cet algorithme étant nécessaire à une mise en train binaire efficace en termes de débit. Néanmoins, pour les bandes de fréquences

élevées ayant un plus grand nombre de coefficients, la contrainte d'un nombre entier de niveaux par échantillon, due à la quantification scalaire, se traduit encore par une granularité trop grossière des résolutions par bande.

La quantification vectorielle permet de s'affranchir de cette contrainte de nombre de niveaux entiers par échantillon et autorise une granularité fine des résolutions disponibles. En revanche, la complexité de la quantification vectorielle limite souvent le nombre de débits disponibles. Par exemple, le codeur de parole multi-débits AMR-NB, basé sur la technique ACELP bien connue, comporte huit débits fixes allant de 12.2 kbit/s à 4.75 kbit/s, chacun ayant un niveau différent de protection contre les erreurs grâce à une distribution différente du débit entre les codages source et canal. Pour chacun des paramètres du codeur ACELP (LSP, retards LTP, gains d'excitation, excitation fixe), des dictionnaires de résolution différente ont été construits. Cependant, le nombre de débits disponibles pour chacun de ces paramètres est limité par la complexité de stockage des quantificateurs vectoriels non algébriques. D'ailleurs, dans le codeur multi-débits AMR-WB comportant neuf débits allant de 6.60 à 23.85 kbit/s, la variation des débits est essentiellement assurée par les dictionnaires d'excitation algébrique qui ne nécessitent pas de stockage. Il y a huit dictionnaires et donc huit débits pour l'excitation fixe tandis que les autres paramètres qui utilisent des dictionnaires stochastiques (LSP, gains, retards absolus et différentiels) n'ont que deux débits possibles.

On indique que les quantificateurs vectoriels stochastiques utilisés dans les codeurs multi-débits AMR sont des quantificateurs vectoriels à structure contrainte (produit cartésien et étages multiples). Une large famille de quantificateurs à débit variable peut en effet être basée sur des quantificateurs vectoriels à structure contrainte tels que les quantificateurs déjà mentionnés à étages multiples, à produits cartésiens, mais aussi les quantificateurs vectoriels en arbre. L'emploi de ces quantificateurs vectoriels en arbre pour le codage à débit variable a fait l'objet de nombreuses études. Le quantificateur vectoriel en arbre binaire fut le premier introduit. Il dérive naturellement de l'algorithme LBG de conception d'un quantificateur vectoriel par séparations ("*splitting*") successives des centroïdes à partir du noeud "*racine*", barycentre de la séquence d'entraînement. Des variantes de quantificateurs vectoriels en arbre ont été proposées en élaguant (méthode de "*pruning*") ou au contraire en ramifiant certains noeuds de l'arbre selon leurs attributs tels que leur distorsion, leur population conduisant à des quantificateurs vectoriels en arbre non binaires et/ou non équilibrés.

Les figures 1a et 1b représentent des quantificateurs vectoriels structurés en arbre. Plus particulièrement, la figure 1a représente un arbre binaire équilibré, tandis que la figure 1b représente un arbre non binaire et non équilibré.

On construit facilement des quantificateurs vectoriels

multi-résolutions à partir d'un quantificateur vectoriel en arbre, en sélectionnant le nombre de noeuds correspondant aux diverses résolutions voulues. La structure hiérarchique en arbre est séduisante et simplifie la procédure de recherche. En revanche, elle implique une recherche sous-optimale et une augmentation importante de la mémoire nécessaire car tous les noeuds de l'arbre depuis le noeud-racine jusqu'aux noeuds terminaux en passant par tous les noeuds des niveaux intermédiaires doivent être stockés. De plus, comme l'ensemble des noeuds d'un dictionnaire de résolution inférieure n'est pas inclus dans les dictionnaires de résolution supérieure, la décroissance de l'erreur de quantification en fonction de l'augmentation du débit du quantificateur vectoriel n'est pas garantie localement.

On sait construire en outre des quantificateurs à résolution variable à partir de codes algébriques, en particulier des quantificateurs vectoriels algébriques imbriqués EAVQ (pour "*Embedded Algebraic Vector Quantizers*") qui utilisent des sous-ensembles de codes sphériques du réseau régulier de Gosset en dimension 8.

Dans le document :
- "*A 16, 24, 32 kbit/s wideband speech codec based on ACELP*" de P. Combescure, J. Schnitzler, K. Fischer, R. Kircherr, C. Lamblin, A. Le Guyader, D. Massaloux, C. Quinquis, J. Stegmann, P. Vary, dans Proceedings IEEE International Conference on Acoustics, Speech, and Signal Processing, Vol. 1, pp 5 -8, 1999,
cette approche de quantification vectorielle algébrique

imbriquée a été étendue à la quantification à dimension variable en utilisant des codes algébriques de différentes dimensions. Même si cette généralisation de la quantification EAVQ permet de quantifier des vecteurs de dimension variable à des résolutions variables, elle présente des inconvénients.

La distribution des vecteurs d'entrée doit être uniforme. Or, adapter la distribution de la source à cette contrainte est une tâche très difficile. La conception de quantificateurs algébriques à partir de réseaux réguliers pose aussi le problème de tronquer et d'ajuster les régions des différents réseaux réguliers pour obtenir les différentes résolutions désirées et ceci pour les différentes dimensions.

La présente invention vient améliorer la situation.

L'un des buts de la présente invention est, de façon générale, de proposer une solution efficace et économique (notamment en mémoire de stockage) au problème de la quantification à débit variable de vecteurs de dimension variable.

Un autre but de la présente invention est, de façon non limitative, de proposer une quantification vectorielle s'adaptant avantageusement au codage et décodage de signaux numériques utilisant une quantification des amplitudes spectrales des codeurs harmoniques et/ou des coefficients de transformée des codeurs fréquentiels, notamment des signaux de parole et/ou audio.

Elle propose à cet effet un dictionnaire comportant des vecteurs-codes de dimension variable et destiné à être utilisé dans un dispositif de codage et/ou décodage en compression de signaux numériques, par quantification vectorielle à débit variable définissant une résolution variable, le dictionnaire comportant :

- d'une part, pour une dimension donnée, des dictionnaires de résolution croissante imbriqués les uns dans les autres,
- et, d'autre part, pour une dimension donnée, une union :

- d'un premier ensemble constitué de vecteurs-codes construits en insérant, dans des vecteurs-codes de dictionnaires de dimension inférieure, des éléments pris dans un ensemble fini de nombres réels selon un jeu fini de règles d'insertion prédéterminées,
- et d'un deuxième ensemble constitué de vecteurs-codes ne pouvant être obtenus par insertion dans des vecteurs-codes de dimension inférieure des éléments dudit ensemble fini selon ledit jeu de règles d'insertion.

Préférentiellement, le jeu de règles d'insertion est élaboré à partir de règles élémentaires consistant à insérer un seul élément de l'ensemble fini de réels en tant que composante à une position donnée d'un vecteur.

Chaque règle élémentaire est préférentiellement définie par un couple de deux entiers positifs représentatifs :

- d'un rang de l'élément dans ledit ensemble fini,
- et d'une position d'insertion.

On comprendra que les règles d'insertion ainsi caractérisées se lisent et se déduisent directement de la structure même du dictionnaire au sens de l'invention.

Bien entendu, de façon purement réversible, on peut définir des règles de suppression consistant à supprimer un ou plusieurs éléments d'un ensemble fini de dimension donnée **N'** pour atteindre une dimension inférieure **N** (**N<N'**).

La présente invention vise aussi un procédé pour former un dictionnaire selon l'invention, dans lequel, pour une dimension donnée :

a) on construit un premier ensemble constitué de vecteurs-codes formés en insérant/supprimant dans des vecteurs-codes de dictionnaires de dimension inférieure/supérieure des éléments pris dans un ensemble fini de nombres réels selon un jeu fini de règles d'insertion/suppression prédéterminées,

b) on construit, pour ladite dimension donnée, un premier dictionnaire, intermédiaire, comportant au moins ledit premier ensemble,

c) et, pour adapter ledit dictionnaire à une utilisation avec au moins une résolution donnée, on construit, à partir du dictionnaire intermédiaire, un second dictionnaire, définitif, par imbrication/simplification de dictionnaires de résolutions croissantes/décroissantes, les dictionnaires de résolutions croissantes étant imbriqués les uns dans les autres du dictionnaire de plus petite résolution jusqu'au dictionnaire de plus grande

résolution.

Bien entendu, on entend par les termes "*imbrication d'un ensemble A dans un ensemble B*", le fait que l'ensemble A est inclus dans l'ensemble B. En outre, on entend par les termes "*simplification d'un ensemble A pour obtenir un ensemble B*", le fait que l'ensemble A inclut l'ensemble B.

En variante ou en complément, on comprendra que les étapes a) et b), d'une part, et l'étape c), d'autre part, peuvent être sensiblement inversées pour adapter ledit dictionnaire à une utilisation avec une dimension donnée **N** de vecteurs-codes.

Dans ce cas :
- à l'étape c), on construit, à partir d'un dictionnaire initial de résolution r_n et de dimension **N'**, un premier dictionnaire, intermédiaire, toujours de dimension **N'** mais de résolution r_N supérieure/inférieure, par imbrication/simplification de dictionnaires de résolutions croissantes/décroissantes, pour atteindre sensiblement la résolution r_N dudit premier dictionnaire,
- à l'étape a), pour atteindre la dimension donnée **N**, on construit un premier ensemble constitué de vecteurs-codes formés en insérant/supprimant, dans des vecteurs-codes du premier dictionnaire de dimension **N'** inférieure/supérieure à ladite dimension donnée **N**, des éléments pris dans un ensemble fini de nombres réels selon un jeu fini de règles d'insertion/suppression prédéterminées,
- et, à l'étape b), suite à une étape éventuelle d'adaptation définitive à la résolution r_N, on construit,

pour ladite dimension donnée **N**, un second dictionnaire, définitif, comportant au moins ledit premier ensemble.

On peut mettre en œuvre l'étape a) par dimensions successives croissantes. Dans ce cas, pour une dimension donnée **N** :

a0) on obtient un dictionnaire initial de dimension initiale **n**, inférieure à ladite dimension donnée **N**,

a1) on construit un premier ensemble constitué de vecteurs-codes de dimension **n+i** et formés en insérant dans des vecteurs-codes du dictionnaire initial des éléments pris dans un ensemble fini de nombres réels selon un jeu fini de règles d'insertion prédéterminées,

a2) on prévoit un deuxième ensemble constitué de vecteurs-codes de dimension **n+i** et ne pouvant être obtenus par insertion dans les vecteurs-codes du dictionnaire initial des éléments dudit ensemble fini avec ledit jeu de règles d'insertion,

a3) on construit un dictionnaire intermédiaire, de dimension **n+i** comportant une union dudit premier ensemble et dudit second ensemble,

et on répète, au plus **N-n-1** fois (auquel cas **i**=1), les étapes a1) à a3), avec ledit dictionnaire intermédiaire en tant que dictionnaire initial, jusqu'à ladite dimension donnée **N**.

On peut aussi mettre en œuvre l'étape a) par dimensions successives décroissantes. Dans ce cas, pour une dimension donnée **N** :

a'0) on obtient un dictionnaire initial de dimension initiale **n**, supérieure à ladite dimension donnée **N**,

a'1) on construit un premier ensemble, de dimension **n-i**, par sélection et extraction de vecteurs-codes possibles de dimension **n-i** dans le dictionnaire de dimension **n**, selon un jeu fini de règles de suppression prédéterminées,
a'2) on prévoit un deuxième ensemble constitué de vecteurs-codes de dimension **n-i**, ne pouvant être obtenus simplement par suppression, dans les vecteurs-codes du dictionnaire initial, des éléments dudit ensemble fini avec ledit jeu de règles de suppression,
a'3) on construit un dictionnaire intermédiaire, de dimension **n-i** comportant une union dudit premier ensemble et dudit second ensemble,
et on répète, au plus **n-N-1** fois (auquel cas **i**=1), les étapes a'1) à a'3), avec ledit dictionnaire intermédiaire en tant que dictionnaire initial, jusqu'à ladite dimension donnée **N**.

Pour obtenir une pluralité de N dictionnaires de dimensions successives 1 à N, on peut combiner les étapes a1) à a3) et les étapes a'1) à a'3), préférentiellement à partir d'un dictionnaire initial de dimension **n** (**n<N**) et par la mise en œuvre répétée des étapes a1) à a3) pour les dimensions **n+1** à **N**, et par la mise en œuvre répétée des étapes a'1) à a'3) pour les dimensions **n-1** à **1**.

On obtient ainsi tout ou partie de **N** dictionnaires dont le dictionnaire de plus grande dimension a pour dimension **N**.

L'ensemble fini et le jeu de règles d'insertion/suppression servant à la construction de dictionnaires de dimensions successives peuvent être

définis :
- a priori, avant de construire le dictionnaire, par analyse d'une source à quantifier,
- ou a posteriori, après la construction de dictionnaires, préférentiellement par imbrication/simplification de dictionnaires de résolutions successives, cette construction étant alors suivie d'une analyse statistique de ces dictionnaires ainsi construits.

On indique que la source à quantifier est préférentiellement modélisée par une séquence d'apprentissage et la définition "*a priori*" de l'ensemble fini et du jeu de règles d'insertion/suppression est préférentiellement effectuée par une analyse statistique de la source. L'ensemble fini précité est préférentiellement choisi par estimation d'une densité de probabilité monodimensionnelle de la source à quantifier.

En combinant des définitions a priori et a posteriori de l'ensemble fini et des règles d'insertion :
- on peut avantageusement choisir, a priori, un premier ensemble et un premier jeu de règles d'insertion/suppression par analyse d'une séquence d'apprentissage, pour former un ou plusieurs dictionnaires intermédiaires,
- on met à jour au moins une partie dudit premier ensemble et/ou dudit premier jeu de règles d'insertion/suppression, par analyse a posteriori desdits un ou plusieurs dictionnaires intermédiaires,
- et, le cas échéant, on met à jour aussi au moins une partie de l'ensemble des vecteur-codes formant lesdits un

ou plusieurs dictionnaires intermédiaires.

Préférentiellement, l'étape c) d'adaptation à une résolution donnée comporte les opérations suivantes, pour atteindre des résolutions croissantes :
c0) on obtient un dictionnaire initial de résolution initiale r_n, inférieure à ladite résolution donnée r_N,
c1) à partir du dictionnaire initial, on construit un dictionnaire intermédiaire de résolution r_{n+1} supérieure à la résolution initiale r_n,
c2) et on répète l'opération c1) jusqu'à atteindre la résolution donnée r_N.

Avantageusement, pour chaque itération de l'opération c1), on prévoit une construction de classes et de centroïdes dans laquelle les centroïdes appartenant au moins aux dictionnaires de résolution supérieure à une résolution courante r_i sont recalculés et mis à jour. En outre, les centroïdes qui appartiennent aux dictionnaires de résolution inférieure à une résolution courante r_i ne sont mis à jour, de préférence, que si les distorsions totales de tous les dictionnaires de résolution inférieure sont décroissantes d'une mise à jour à l'autre.

En complément ou en variante, l'étape c) comporte les opérations suivantes, maintenant pour atteindre des résolutions décroissantes :
c'0) on obtient un dictionnaire initial de résolution initiale r_n, supérieure à ladite résolution donnée r_N,
c'1) à partir du dictionnaire initial, on construit un dictionnaire intermédiaire de résolution r_{n-1} inférieure à

la résolution initiale r_n, par partition du dictionnaire initial en plusieurs sous-ensembles ordonnés selon un critère prédéterminé, et

c'2) on répète l'opération c'1) jusqu'à atteindre la résolution donnée r_N.

Avantageusement, cette partition peut utiliser la composition partielle par extension contrôlée au sens des étapes a) et b), en utilisant une partie au moins des règles d'insertion/suppression mises en œuvre.

Pour obtenir une pluralité de **N** dictionnaires successifs de résolutions respectives r_1 à r_N, à partir d'un dictionnaire initial de résolution r_n intermédiaire entre les résolutions r_1 et r_N, on peut mettre avantageusement en œuvre une répétition de l'étape c1) pour les résolutions croissantes r_{n+1} à r_N, et de l'étape c'1) pour les résolutions décroissantes r_{n-1} à r_1.

On comprendra que l'ensemble fini et le jeu des règles d'insertion/suppression peuvent avantageusement être choisis par une étude, a posteriori, d'une statistique des dictionnaires de différentes résolutions et dimensions ainsi obtenus, pour former un dictionnaire au sens de l'invention, de dimensions et de résolutions souhaitées.

Selon l'un des avantages procurés par la présente invention, le stockage en mémoire nécessaire pour la mise en œuvre du codage/décodage peut être considérablement réduit. En effet, de façon avantageuse, on stocke dans une mémoire, une fois pour toutes, ledit jeu de règles

d'insertion/suppression, identifiées chacune par un indice, et, pour une dimension donnée :
- ledit deuxième ensemble constitué de vecteurs-codes ne pouvant être obtenus par application des règles d'insertion/suppression aux vecteurs-codes de dimension inférieure/supérieure à la dimension donnée,
- ainsi qu'au moins une table de correspondance permettant de reconstituer un vecteur-code quelconque du dictionnaire de dimension donnée, en utilisant les indices des règles d'insertion/suppression et des indices identifiant des éléments dudit second ensemble.

On évite ainsi le stockage complet du dictionnaire pour ladite dimension donnée, en stockant simplement les éléments dudit second ensemble et des liens dans la table de correspondance pour accéder à ces éléments et aux règles d'insertion/suppression associées.

Ainsi, on comprendra que, pour une dimension donnée, le deuxième ensemble précité peut être avantageusement constitué de "*deuxièmes*" sous-ensembles de dimensions inférieures à ladite dimension donnée.

Dans une réalisation, le mécanisme d'insertion/suppression lui-même peut être stocké en tant que routine de programme, tandis que les paramètres d'insertion/suppression, pour une règle d'insertion/suppression donnée, peuvent être stockés dans une table de correspondance générale (en principe différente de la table de correspondance précitée), en combinaison avec l'indice de cette règle

d'insertion/suppression donnée.

Préférentiellement, les tables de correspondances sont élaborées au préalable, pour chaque indice d'un vecteur-code d'un dictionnaire de dimension donnée pouvant être reconstruit à partir d'éléments d'indices courants dans le second ensemble de dimension courante, par une tabulation de trois valeurs scalaires entières représentant :
- une dimension courante dudit second ensemble,
- un indice courant d'élément du second ensemble, et
- un indice de règle d'insertion/suppression,
cette règle d'insertion/suppression au moins contribuant à reconstituer ledit vecteur-code du dictionnaire de dimension donnée, en appliquant l'insertion/suppression à l'élément correspondant audit indice courant et à ladite dimension courante.

Ces dernières caractéristiques peuvent avantageusement être mises en œuvre dans un procédé de codage/décodage en compression, comme décrit ci-après.

A ce titre, la présente invention vise aussi une utilisation du dictionnaire selon l'invention et obtenu par la mise en œuvre des étapes ci-avant, pour le codage ou le décodage en compression de signaux numériques, par quantification vectorielle à débit variable définissant une résolution variable. En particulier, on recherche le vecteur-code le plus proche voisin d'un vecteur d'entrée $y = (y_0, \ldots, y_k, \ldots, y_{j-1})$ dans un dictionnaire de dimension donnée j. Cette utilisation met alors en œuvre les étapes suivantes :

*CO1) pour un indice courant dudit vecteur-code recherché, reconstitution au moins partielle d'un vecteur-code d'indice correspondant audit indice courant, au moins par lecture préalable des indices figurant dans les tables de correspondance et, le cas échéant, d'un élément du deuxième ensemble, permettant d'élaborer ledit dictionnaire,

le procédé se poursuivant par des étapes de codage/décodage proprement dites, comportant :

*CO2) au moins au codage, calcul d'une distance entre le vecteur d'entrée et le vecteur-code reconstitué à l'étape CO1),

*CO3) au moins au codage, répétition des étapes CO1) et CO2), pour tous les indices courants dans ledit dictionnaire,

CO4) au moins au codage, identification de l'indice du vecteur-code au moins partiellement reconstitué dont la distance avec le vecteur d'entrée, calculée au cours de l'une des itérations de l'étape CO2), est la plus petite, et

CO5) au moins au décodage, détermination du plus proche voisin du vecteur d'entrée y en tant que vecteur-code dont l'indice a été identifié à l'étape CO4).

Comme indiqué ci-avant, on rappelle que le "*deuxième*" ensemble précité est préférentiellement constitué de "*deuxièmes*" sous-ensembles de dimensions inférieures à une dimension donnée du deuxième ensemble.

Dans un mode de réalisation particulier, l'étape CO1), au moins au décodage, comporte :

CO11) la lecture, dans les tables de correspondance, d'indices représentatifs de liens vers ledit deuxième ensemble et vers les règles d'insertion et incluant :

- l'indice d'une dimension courante d'un sous-ensemble dudit deuxième ensemble,
- l'indice courant d'un élément dudit sous-ensemble,
- et l'indice de la règle d'insertion appropriée pour la construction du vecteur-code du dictionnaire de dimension donnée, à partir dudit élément,

CO12) la lecture, dans ledit sous-ensemble identifié par sa dimension courante, dudit élément identifié par son indice courant,

CO13) la reconstitution complète du vecteur-code à ladite dimension donnée en appliquant audit élément lu à l'étape CO12) la règle d'insertion appropriée et identifiée par son indice lu à l'étape CO11).

Dans un mode de réalisation particulier, au codage,

* l'étape CO1) comporte :

CO11) la lecture, dans les tables de correspondance, d'indices représentatifs de liens vers ledit deuxième ensemble et vers les règles d'insertion et incluant :

- l'indice d'une dimension courante d'un sous-ensemble dudit deuxième ensemble,
- l'indice courant d'un élément dudit sous-ensemble,
- et l'indice de la règle d'insertion appropriée pour la construction du vecteur-code du dictionnaire de dimension donnée, à partir dudit élément,

CO12) la lecture, dans le sous-ensemble identifié par sa dimension courante, dudit élément identifié par son indice courant,

* à l'étape CO2), on calcule ladite distance en fonction d'un critère de distorsion estimé en fonction de :
- de ladite règle d'insertion,
- et dudit élément.

Ainsi, on peut ne prévoir qu'une reconstruction partielle du vecteur-code à ladite dimension donnée à l'étape CO1), en réservant la reconstruction complète simplement au décodage.

Dans un mode de réalisation avantageux, on prévoit en outre une propriété structurante supplémentaire selon une union de codes à permutation, et on exploite en outre une indexation de cette union de codes à permutation dans la mise en œuvre des étapes suivantes :
CP1) à partir d'un signal d'entrée, on forme un vecteur d'entrée $y = (y_0, \ldots, y_k, \ldots, y_{j-1})$ défini par son vecteur absolu $|y| = (|y_0|, \cdots, |y_k|, \cdots, |y_{j-1}|)$ et par un vecteur signe $\varepsilon = (\varepsilon_0, \ldots, \varepsilon_k, \ldots, \varepsilon_{j-1})$ avec $\varepsilon_k = \pm 1$,
CP2) on range les composantes du vecteur $|y|$ par valeurs décroissantes, par permutation, pour obtenir un vecteur leader $|\tilde{y}|$,
CP3) on détermine, parmi les vecteurs leaders du dictionnaire D^j_i de dimension j, un plus proche voisin $x^{j'}$ du vecteur leader $|\tilde{y}|$,
CP4) on détermine un index du rang dudit plus proche voisin $x^{j'}$ dans le dictionnaire D^j_i,
CP5) et l'on applique une valeur effective de codage/décodage au vecteur d'entrée, qui est fonction dudit index déterminé à l'étape CP4), de la ladite

permutation déterminée à l'étape CP2) et dudit vecteur signe déterminé à l'étape CP1).

Selon un autre aspect avantageux de l'invention, pour le codage/décodage et éventuellement pour la construction du ou des dictionnaires, on prévoit de stocker les tables de correspondance et les éléments du deuxième ensemble précités, notamment dans une mémoire d'un dispositif de codage/décodage en compression.

A ce titre, la présente invention vise aussi un tel dispositif de codage/décodage.

La présente invention vise aussi un produit programme d'ordinateur destiné à être stocké dans une mémoire d'une unité de traitement, notamment d'un ordinateur ou d'un terminal mobile, ou sur un support mémoire amovible et destiné à coopérer avec un lecteur de l'unité de traitement, ce programme comportant des instructions pour la mise en œuvre du procédé de construction de dictionnaires ci-avant.

La présente invention peut aussi viser un programme de ce type, en particulier un produit programme d'ordinateur destiné à être stocké dans une mémoire d'une unité de traitement, notamment d'un ordinateur ou d'un terminal mobile intégrant un dispositif de codage/décodage, ou sur un support mémoire amovible et destiné à coopérer avec un lecteur de l'unité de traitement, ce programme comportant alors des instructions pour la mise en œuvre de l'application au codage/décodage en compression ci-avant.

D'autres caractéristiques et avantages de l'invention apparaîtront à l'examen de la description détaillée ci-après, et des dessins annexés sur lesquels, outre les figures 1a et 1b décrites ci-avant :

- la figure 2a illustre la propriété d'imbrication d'un dictionnaire au sens de l'invention, pour une dimension donnée N,
- la figure 2b illustre la propriété de composition partielle par extension contrôlée d'un dictionnaire au sens de l'invention,
- la figure 3 illustre l'imbrication des dictionnaires en fonction des résolutions croissantes,
- la figure 4 illustre la composition de vecteurs-codes d'un dictionnaire à partir de vecteurs-codes de dictionnaires de dimensions inférieures et de règles d'insertion,
- la figure 5 illustre la construction selon les résolutions croissantes de dictionnaires imbriqués sans remise à jour des dictionnaires de résolution inférieure,
- la figure 6 illustre le schéma bloc du codeur "*TDAC*",
- les figures 7a à 7g représentent, pour le codeur TDAC en bande élargie utilisant un quantificateur vectoriel au sens de l'invention, des tableaux illustrant respectivement :

* une découpe en 32 bandes (fig.7a),
* les résolutions par dimension (fig.7b),
* le gain en mémoire apporté par la propriété d'imbrication (fig.7c),
* le gain en mémoire apporté par les deux propriétés

d'imbrication et d'extension contrôlée (fig.7d),
* le gain en mémoire apporté par les deux propriétés structurantes en fonction de la dimension et du débit, respectivement, par rapport à la taille mémoire nécessaire au stockage d'un dictionnaire sans utiliser ces deux propriétés (fig.7e),
* les premiers leaders de l'ensemble L^0 en dimensions 1, 2 et 3 (fig.7f), et
* les leaders des codes en permutation des dictionnaires en dimension 3 (fig.7g),
- les figures 8a et 8b représentent, pour le codeur TDAC en bande FM, des tableaux illustrant respectivement :
* une découpe en 52 bandes (fig.8a), et
* les résolutions par dimension (fig.8b).

On se réfère tout d'abord aux figures 2a et 2b qui illustrent les deux propriétés principales d'un dictionnaire D_i^N au sens de la présente invention.

Sur la figure 2a, pour une dimension donnée N, des dictionnaires D_1^N, D_2^N, ..., D_i^N de résolutions respectives croissantes r_1, r_2, ..., r_i sont imbriqués les uns dans les autres. Ainsi, le dictionnaire D_i^N de résolution maximale r_i peut permettre de déterminer un dictionnaire D_j^N de résolution r_j inférieure ($j<i$), comme on le verra plus loin. Cette première propriété, notée PR, est nommée ci-après "*propriété d'imbrication*".

En se référant maintenant à la figure 2b, tout dictionnaire D_i^N d'une dimension donnée N et de résolution r_i est l'union de deux ensembles disjoints :

- un premier ensemble $D'_i{}^N$ étant constitué de vecteurs-codes Y^N construits (flèche F3) en insérant dans des vecteurs-codes Y^{N-1} des dictionnaires $D_i{}^{N-1}$ de dimension inférieure N-1 des éléments x_j pris (flèche F2) dans un ensemble fini A de nombres réels selon un jeu fini de règles d'insertion $\{R_m\}$, une règle d'insertion R'(j,k) déterminant les éléments x_j à insérer (flèche F1) et la façon de les insérer (par exemple à une position k du vecteur Y^N en construction),
- et un deuxième ensemble $\overline{D}'^N_i$ étant constitué de vecteurs Y' ne pouvant être obtenus en insérant dans des vecteurs-codes de dimension inférieure des éléments de cet ensemble fini selon le jeu de règles d'insertion précité.

Cette deuxième propriété, notée PD, est nommée ci-après "*propriété de composition partielle par extension contrôlée*".

Sur les figures 2a et 2b et dans le résumé de l'invention ci-avant, les indices en résolution et/ou en dimension commencent, à titre d'exemple, de l'entier **1** jusqu'à un entier donné (**i**, **n**, ou **N** selon le cas). L'homme du métier de la programmation, notamment en langage C++, comprendra que ces indices peuvent partir plutôt de **0** et atteindre **i-1**, **n-1**, ou **N-1**, selon le contexte. C'est ainsi que sur l'exemple de la figure 3 qui sera décrit plus loin, la plus grande résolution atteinte est N_j-1 en partant de 0.

On décrit ci-après un procédé de construction d'un

dictionnaire possédant les deux propriétés structurantes PR et PD, en particulier des algorithmes de construction de ces dictionnaires ainsi structurés. Les liens induits par les deux propriétés structurantes sont avantageusement exploités pour développer des algorithmes de construction de tels dictionnaires en adaptant les algorithmes itératifs de construction couramment utilisés et décrits ci-avant tels que le "GLA" ou le "SKA".

De façon générale, on indique que :

- des dictionnaires de résolutions différentes et de même dimension, liés entre eux, sont construits successivement en utilisant la propriété d'imbrication PR,
- en complément ou en variante, on construit des dictionnaires de dimensions différentes liés entre eux par la propriété PD de composition partielle par extension contrôlée,
- et on obtient ainsi des dictionnaires de différentes dimensions et résolutions possédant les deux propriétés structurantes PD et PR.

De façon générale, pour construire des dictionnaires imbriqués en résolution croissante pour une dimension donnée (PR), trois approches de construction sont proposées.

Une première approche consiste à construire les dictionnaires selon les résolutions croissantes (de la plus petite résolution jusqu'à la résolution maximale).

Une deuxième approche consiste inversement à construire les dictionnaires selon les résolutions décroissantes (de la résolution maximale jusqu'à la plus petite résolution).

Une troisième approche consiste à construire les dictionnaires à partir d'un dictionnaire de résolution intermédiaire en faisant décroître les résolutions jusqu'à la résolution minimale et en les faisant augmenter jusqu'à la résolution maximale. Cette méthode est particulièrement intéressante lorsque la résolution nominale du quantificateur vectoriel de résolution variable est la résolution intermédiaire précitée.

La propriété PR d'imbrication des dictionnaires, pour une dimension j se traduit finalement par :

$$D_0^j \subset D_1^j \subset \cdots D_i^j \subset D_{i+1}^j \cdots \subset D_{N_j-1}^j$$

En notant :

- N_j le nombre de résolutions (ou de débits possibles dans un codeur à débit variable) en dimension j,
- l'ensemble des résolutions en dimension j $R_j = \left\{ r_0^j, r_1^j, \cdots, r_i^j, r_{i+1}^j, \cdots, r_{N_j-1}^j \right\}$,

  avec $r_0^j < r_1^j < \cdots < r_i^j < r_{i+1}^j < \cdots < r_{N_j-1}^j$
- D_i^j le dictionnaire de dimension j, de résolution r_i^j
- T_i^j la taille du dictionnaire de résolution $r_i^j \left(T_i^j = 2^{jr_i^j} \text{ i.e. } r_i^j = \frac{1}{j} \log_2 T_i^j \right)$

La figure 3 illustre l'imbrication des dictionnaires en fonction des résolutions croissantes.

L'organigramme de l'algorithme de construction selon les résolutions croissantes sans remise à jour des dictionnaires de résolution inférieure est donné à la figure 5.

En se référant à la figure 5, on construit d'abord le dictionnaire D_0^j de plus faible résolution, suite aux étapes d'initialisation 51 et 52 où l'on fixe d'abord i=0 et l'indice d'itération de boucle iter=0. Puis le dictionnaire D_0^j de plus faible résolution étant fixé, le dictionnaire de résolution immédiatement supérieure D_1^j est construit à l'aide d'une variante d'un algorithme classique de construction, décrit ci-après. Le procédé est ensuite itéré jusqu'à construire le dictionnaire de résolution maximale $D_{N_j-1}^j$.

Ainsi, à l'étape 53 où, par un processus itératif, on cherche à construire un dictionnaire D_i^j à partir d'un dictionnaire initial $D_i^j(0)$, formé en ajoutant $(T_i^j - T_{i-1}^j)$ vecteurs au dictionnaire D_{i-1}^j de résolution inférieure r_{i-1}.

L'algorithme de construction des classes 54 est identique à un algorithme classique mais l'algorithme de construction des T_i^j centroïdes 55 est modifié. En effet, les $(T_i^j - T_{i-1}^j)$ centroïdes n'appartenant pas aux dictionnaires de résolution inférieure sont recalculés et

mis à jour, tandis que les (T_{i-1}^{j}) centroïdes des dictionnaires de résolution inférieure ne sont pas remis à jour. Une variante autorise la remise à jour des centroïdes des dictionnaires des résolutions inférieures dans le cas où les distorsions totales de tous les dictionnaires de résolution inférieure décroissent ou restent constantes. Dans ce cas, les dictionnaires de résolutions inférieures sont modifiés en conséquence.

L'indice de boucle iter est ensuite incrémenté (étape 56) jusqu'à un nombre Niter(i,j) dépendant de la $i^{ème}$ résolution et de la dimension j (test 57). Une fois que la résolution souhaitée N_j est atteinte (test 58), on obtient le dictionnaire à cette résolution N_j (étape de fin 59), et donc l'ensemble des dictionnaires D_i^j de résolution r_i, pour i allant de 1 à N_j.

Pour construire les dictionnaires selon les résolutions décroissantes, on construit d'abord le dictionnaire de plus haute résolution. Puis celui-ci étant fixé, on effectue une partition de celui-ci en plusieurs sous-ensembles qu'on ordonne selon un certain critère. Plusieurs critères peuvent servir à ordonner la partition. On peut par exemple ordonner les sous-ensembles selon leur cardinal, leur sollicitation dans la séquence d'apprentissage (c'est-à-dire le cardinal de leurs régions de quantification), leur contribution à la distorsion totale ou plus précisément à la décroissance de cette distorsion. On peut évidemment combiner divers critères et pondérer leur importance respective. De même, la partition du dictionnaire peut être effectuée de diverses manières:

de la partition élémentaire (un élément dans chaque sous-ensemble) à une partition plus élaborée. Cette partition ordonnée est à la base de la construction des dictionnaires imbriqués par union progressive de ses classes ordonnées.

Préférentiellement, la partition peut s'appuyer sur la propriété PD de composition partielle par extension contrôlée en regroupant les éléments basés sur l'extension d'un même vecteur-code à partir d'un sous-ensemble du jeu de règles d'insertion (éventuellement égal cet ensemble lui-même).

Il faut noter que l'on peut faire plusieurs itérations en alternant les différentes méthodes. Par exemple, on construit des dictionnaires imbriqués selon la méthode par résolutions croissantes puis on applique la méthode par résolutions décroissantes. En combinant les deux procédés ci-avant, des dictionnaires imbriqués en résolution sont construits à partir d'un dictionnaire de résolution intermédiaire r_i. Cet $i^{ème}$ dictionnaire est donc d'abord construit. Puis, à partir de ce dictionnaire, on construit les dictionnaires de résolution inférieure à l'aide du second procédé par résolutions décroissantes et les dictionnaires de résolutions supérieures à l'aide du premier procédé par résolutions croissantes.

De manière générale, on propose aussi trois approches pour construire des dictionnaires de différentes dimensions par composition partielle par extension contrôlée (propriété PD).

Une première approche consiste à augmenter les dimensions. Une autre approche consiste à les diminuer. Enfin, une dernière approche consiste à commencer par construire le dictionnaire d'une dimension intermédiaire et construire par augmentation et diminution successives de la dimension les dictionnaires de dimensions supérieures et inférieures. La composition partielle par extension contrôlée a conduit à mettre au point des procédures de détermination de l'ensemble fini de réels et du jeu de règles d'insertion que l'on verra ci-après. On indique simplement ici que, préférentiellement, la proportion d'éléments "*étendus*" (nombre d'éléments du premier ensemble par rapport au cardinal du dictionnaire) est croissante avec la dimension, ce qui permet de réduire le coût de stockage du deuxième ensemble, augmentant avec la dimension. Cette proportion peut être fixée a priori par les contraintes de complexité de l'application (mémoire/puissance de calcul) ou laissée "*libre*". Dans ce dernier cas, l'algorithme de construction favorise avantageusement les éléments du premier ensemble comprenant les éléments obtenus par extension contrôlée, comme on le verra ci-après.

Ainsi, la deuxième propriété PD de composition partielle par extension contrôlée se traduit finalement par :

$$D_i^j = D'^j_i \bigcup \overline{D'^j_i}$$

en notant :

- D'^j_i l'ensemble des vecteurs-codes de D_i^j qui peuvent être obtenus en insérant dans des vecteurs-codes des

dictionnaires des dimensions inférieures des éléments pris d'un ensemble fini A de $\mathbb{R}$ selon un jeu de règles d'insertion $\{R_m\}$,

- $\overline{D'^j_i}$ son complémentaire dans D^j_i, ensemble des vecteurs-codes de D^j_i ne pouvant être obtenus en insérant dans des vecteurs-codes de dimension inférieure des éléments de A selon le jeu de règles d'insertion $\{R_m\}$.

On décrit ci-après un exemple de règles d'insertion pour vérifier la seconde propriété PD.

Tout d'abord, on définit un jeu de règles élémentaires d'insertion : chaque règle élémentaire consiste à insérer un et un seul élément de l'ensemble fini de réels A comme composante à une position donnée d'un vecteur. Chaque règle élémentaire est donnée par un couple de deux entiers positifs, l'un donnant le rang de l'élément dans l'ensemble fini et l'autre la position d'insertion. A partir de ce jeu de règles élémentaires, on peut composer une règle quelconque, plus élaborée, d'insertion de composantes.

Bien entendu, de façon purement réversible, on peut définir des règles de suppression consistant à supprimer un ou plusieurs éléments d'un ensemble fini de dimension donnée N pour atteindre une dimension inférieure N-n.

Pour définir une règle d'insertion, on note alors :

- N_a le cardinal de A et a_i son $i^{ème}$ élément :

$A = \{a_0, a_1, \ldots, a_i \ldots, a_{Na-1}\}$,

- $R'(i_m, p_m)$ la règle d'insertion élémentaire qui consiste à insérer a_{i_m} en position p_m.

Ainsi, si la dimension maximale est j_{max}, le nombre de règles élémentaires possibles est $N_a * j_{max}$. Par exemple, pour N_a=2 et j_{max}=3, on compte en tout six règles élémentaires possibles:

$R'(0,0)$: insérer a_0 en position 0,
$R'(1,0)$: insérer a_1 en position 0,
$R'(0,1)$: insérer a_0 en position 1,
$R'(1,1)$: insérer a_1 en position 1,
$R'(0,2)$: insérer a_0 en position 2,
$R'(1,2)$: insérer a_1 en position 2

La composée des règles $R'(0,0)$ et $R'(0,1)$ donne la règle : insérer a_0 en positions 0 et 1. Elle permet d'obtenir ainsi un vecteur-code de dimension j+2 à partir d'un vecteur-code de dimension j.

La composée des règles $R'(1,0)$ et $R'(0,2)$ donne la règle : insérer a_1 en position 0 et a_0 en position 2. Elle permet aussi d'obtenir un vecteur-code de dimension j+2 à partir d'un vecteur-code de dimension j.

Plus généralement, on note $R(n, \{(i_m, p_m)\} m = 0, n = 1)$ la composée des n règles élémentaires $R'(i_m, p_m)$ (de m=0 à n-1), qui permet d'obtenir un vecteur-code de dimension j+n à partir d'un vecteur-code de dimension j. Il faut noter que les i_m ne sont pas nécessairement différents, par contre les n positions p_m sont distinctes. Préférentiellement, on

agence les positions p_m en ordre croissant, soit :

$$p_0<p_1\ldots<p_m\ldots<p_{n-1}.$$

La figure 4 illustre la composition de vecteurs-codes d'un dictionnaire à partir de vecteurs-codes de dictionnaires de dimensions inférieures et de règles d'insertion.

On prévoit aussi plusieurs modes de réalisation pour construire des dictionnaires de différentes dimensions, unions de deux ensembles disjoints, un premier ensemble étant constitué de vecteurs-codes construits en insérant à des vecteurs-codes des dictionnaires des dimensions inférieures des éléments pris d'un ensemble fini de nombres réels selon un jeu de règles d'insertion, un deuxième ensemble étant constitué de vecteurs ne pouvant être obtenus en insérant aux vecteurs-codes de dimension inférieure des éléments de cet ensemble fini de nombres réels selon ce jeu de règles d'insertion.

Le premier ensemble nécessite la détermination de l'ensemble fini de réels (c'est-à-dire son cardinal et ses valeurs) ainsi que du jeu de règles d'insertion.

La construction de cet ensemble fini et l'élaboration du jeu de règles d'insertion sont effectuées :

- soit "*a priori*" : l'ensemble fini et le jeu de règles d'insertion sont déterminés avant de construire les dictionnaires. Ce choix s'appuie préférentiellement sur une analyse des statistiques de la source à quantifier,

modélisée par exemple par une séquence d'apprentissage. Par exemple, le choix de l'ensemble fini peut s'appuyer sur la densité de probabilité monodimensionnelle de la source (ou son histogramme);

- soit *"a posteriori"* : on construit d'abord les dictionnaires imbriqués en résolution pour toutes les dimensions sans imposer de suivre la règle de composition partielle par extension contrôlée. Le choix de l'ensemble fini et du jeu des règles d'insertion est ensuite effectué par une étude de la statistique de ces dictionnaires *"initiaux"*.

Les deux solutions "*a priori*" ou "*a posteriori*" peuvent être utilisées successivement et/ou combinées. Par exemple, un premier ensemble et un premier jeu de règles d'insertion peuvent être choisis par une analyse de la séquence d'apprentissage, puis après une première construction des dictionnaires, une analyse de ces dictionnaires peut conduire à une mise à jour totale ou partielle de l'ensemble A et/ou du jeu de règles d'insertion.

Il faut aussi noter que l'ensemble fini et/ou le jeu de règles d'insertion peuvent être dépendants ou non des dimensions. On peut alors déterminer un jeu et/ou un ensemble spécifique pour chaque couple de dimensions (j,j'), ou un jeu et/ou un ensemble spécifique par différence de dimension, ou déterminer un ensemble global. Là encore, le choix est fait a priori ou après analyse statistique de la séquence d'apprentissage et/ou des

dictionnaires.

Pour construire les dictionnaires selon les dimensions croissantes, on construit d'abord le dictionnaire de plus faible dimension par une méthode classique de conception de quantification vectorielle, comme indiqué ci-avant. Puis, ce dictionnaire étant construit, le dictionnaire de dimension immédiatement supérieure est construit à l'aide d'une variante d'un algorithme classique de construction. A partir du dictionnaire de dimension inférieure, on compose tous les vecteurs-codes initiaux possibles en appliquant les règles d'insertion, on complète éventuellement ce dictionnaire par des vecteurs-codes "*libres*" (c'est-à-dire ceux qui ne peuvent pas être obtenus par extension). Il faut noter que la taille de ce dictionnaire initial peut être supérieure à la taille désirée. A partir du dictionnaire initial, une variante d'un algorithme itératif de construction d'un quantificateur vectoriel est alors appliquée. On construit des classes par quantification de la séquence d'apprentissage et l'on met à jour des centroïdes en respectant la contrainte d'extension contrôlée pour les vecteurs-codes du premier ensemble. Pour ces vecteurs-codes du premier ensemble, on peut soit ne pas recalculer les composantes obtenues par insertion, soit recalculer toutes les composantes et modifier les vecteurs-codes ainsi obtenus pour faire réapparaître les composantes obtenues par les règles d'insertion. On élimine aussi les classes vides si la taille du dictionnaire est supérieure à la taille voulue. Si à la fin de l'algorithme, la taille du dictionnaire est supérieure à la résolution voulue, une

procédure de classement des éléments du dictionnaire est appliquée pour ne retenir que les premiers vecteurs-codes. L'algorithme itératif est éventuellement relancé. On passe ensuite à la construction du dictionnaire de la dimension supérieure, le dictionnaire initial est alors construit par extension contrôlée à partir des deux dictionnaires des deux plus petites dimensions et complété par des vecteurs-codes *"libres"*, puis on applique la variante de l'algorithme itératif de construction d'un quantificateur vectoriel. Le procédé est ensuite itéré, jusqu'à construire le dictionnaire de dimension maximale.

En variante, pour construire les dictionnaires selon les dimensions décroissantes, on construit d'abord le dictionnaire de plus grande dimension. Puis, celui-ci étant fixé, on extrait les vecteurs-codes de dimension inférieure possibles. Avantageusement, la procédure d'extraction est facilitée en modifiant les vecteurs-codes des dimensions supérieures pour faire apparaître des éléments de A comme composantes de ces vecteurs-codes.

Dans une variante complémentaire, on effectue avantageusement plusieurs itérations en alternant les deux constructions selon les dimensions croissantes, d'une part, et selon les dimensions décroissantes, d'autre part.

Pour faciliter la procédure d'extension contrôlée, l'invention peut en plus procéder à une transformation des composantes des vecteurs-codes. Un exemple de transformation est une quantification scalaire à haute résolution. Il est intéressant de construire des

"dictionnaires" de dimensions inférieures même si ces dimensions ne sont pas utilisées directement par la quantification vectorielle. Par exemple, on peut commencer par la dimension 1 même si la quantification scalaire n'est pas utilisée. De même, il peut aussi être intéressant de construire des dictionnaires des dimensions intermédiaires. Ces *"dictionnaires"* sont d'ailleurs avantageusement utilisés par la procédure d'extension contrôlée pour réduire la complexité de stockage et de calculs.

On indique en outre qu'en combinant judicieusement des algorithmes de construction de dictionnaires par imbrication en résolution (PR) avec des algorithmes de construction par composition partielle par extension contrôlée (PD), plusieurs procédés de construction peuvent être développés. Il faut noter que les algorithmes étant itératifs, différentes techniques peuvent être alternées. Par exemple, on commence par construire le dictionnaire de résolution maximale pour la plus petite dimension, on en déduit les dictionnaires imbriqués en résolutions décroissantes (propriété PR), puis on construit le dictionnaire de résolution maximale pour la dimension immédiatement supérieure grâce à la propriété PD, pour cette dimension, on construit les dictionnaires imbriqués en résolution et on itère jusqu'à obtenir les dictionnaires (imbriqués en résolutions) de dimension maximale.

Une construction préférentielle est utilisée dans le mode de réalisation décrit ci-après qui combine les techniques

de construction de dictionnaire selon les dimensions croissantes et les résolutions décroissantes pour construire l'ensemble des dictionnaires $\left\{D_i^j\right\}_{i=0,\cdots,N_j-1, j=j_{\min},\cdots,j_{\max}}$.

On décrit ci-après le codage/décodage en compression de signaux numériques (audio, vidéo, ...), utilisant des dictionnaires au sens de l'invention, en particulier les algorithmes de codage et de décodage qui exploitent la structure des dictionnaires (imbrication et composition partielle par extension contrôlée). De manière générale, on comprendra qu'une optimisation du compromis mémoire/calculs au codeur et/ou au décodeur est menée selon les contraintes de l'application.

A titre d'exemple, on considère ci-après le codeur audio nommé "*codeur TDAC*", utilisé pour coder des signaux audio numériques échantillonnés à 16 kHz (en bande élargie). Ce codeur est un codeur par transformée qui peut fonctionner à différents débits. En particulier, le débit peut être fixé avant l'établissement de la communication ou varier de trame à trame en cours de communication.

La figure 6 donne le schéma bloc de ce codeur TDAC. Un signal audio x(n) limité en bande à 7 kHz et échantillonné à 16 kHz est découpé en trames de 320 échantillons (20 ms). Une transformée en cosinus discrète modifiée 61 est appliquée sur des blocs du signal d'entrée de 640 échantillons avec un recouvrement de 50% (c'est-à-dire un rafraîchissement de l'analyse MDCT toutes les 20 ms). On

limite le spectre obtenu y(k) à 7225 Hz en mettant à zéro les 31 derniers coefficients (seuls les 289 premiers coefficients sont différents de 0). Une courbe de masquage est déterminée par le module de masquage 62 qui effectue ensuite une mise à zéro des coefficients masqués. Le spectre est divisé en trente-deux bandes de largeurs inégales. Les éventuelles bandes masquées sont déterminées en fonction des coefficients transformés du signal x(n). Pour chaque bande du spectre, l'énergie des coefficients MDCT est calculée (on parle de facteurs d'échelle). Les trente-deux facteurs d'échelle constituent l'enveloppe spectrale du signal qui est ensuite quantifiée, codée et transmise dans la trame (bloc 63). Cette quantification et ce codage utilisent un codage de Huffman. Le nombre variable de bits restant après la quantification de l'enveloppe spectrale à débit variable est alors calculé. Ces bits sont distribués pour la quantification vectorielle 65 des coefficients MDCT du spectre. L'enveloppe spectrale déquantifiée sert à calculer l'ensemble des seuils de masquage par bande, cette courbe de masquage déterminant l'allocation dynamique des bits 64. Le calcul de cette courbe de masquage bande par bande et à partir de l'enveloppe spectrale quantifiée évite la transmission d'informations auxiliaire relatives à l'allocation binaire. En effet, le décodeur calcule l'allocation dynamique des bits de manière identique au codeur. Les coefficients MDCT sont normalisés par les facteurs d'échelle déquantifiés de leur bande puis ils sont quantifiés par des quantificateurs vectoriels de dimension et débit variables. Finalement, le train binaire est construit par multiplexage 66 des informations sur

l'enveloppe spectrale et ces coefficients normalisés par bande codés et transmis en trame. On indique que les références 67 et 68 sur la figure 6 correspondent à des étapes connues en soi de détection d'un signal x(n) voisé ou non voisé, et de détection de tonalité (détermination de fréquences tonales), respectivement.

On décrit ci-après les quantificateurs vectoriels à débit variable par bandes de largeurs inégales des coefficients MDCT dans le codeur TDAC. La quantification des coefficients MDCT normalisés par bande utilise en particulier des dictionnaires construits selon l'invention. La découpe en bandes de largeurs inégales conduit en effet à des vecteurs de différentes dimensions. Le tableau de la figure 7a qui donne la découpe en bande utilisée indique aussi la dimension résultante du vecteur des coefficients, c'est-à-dire le nombre de coefficients indiqué par la troisième colonne.

Le nombre variable de bits restant après le codage d'Huffman de l'enveloppe spectrale est alloué dynamiquement aux différentes bandes. Le tableau de la figure 7b donne les nombres de résolutions N_j et les ensembles des débits par bandes $j*R_j$ (donc les valeurs des résolutions par bande) pour les dimensions j, pour j allant de 1 à 15. On notera que pour exploiter avantageusement la propriété structurante de composition partielle par extension contrôlée, on a construit des quantificateurs vectoriels en dimensions 1, 2, 6, 11, qui, pourtant, ne correspondent à aucune largeur de bande, mais dont les éléments servent à composer des vecteurs-codes de

dimension plus élevée. On constate aussi la finesse de la granularité des résolutions même pour des dimensions élevées.

La mise à zéro des coefficients masqués dans le module 62 conduit à choisir, lors de l'analyse des coefficients MDCT normalisés, comme ensemble de départ A={0} et comme jeu de règles d'insertion toutes les composées possibles des règles élémentaires d'insertion. Cela revient ici à insérer des zéros à une position quelconque.

Toutefois, une analyse plus fine impose aux dictionnaires une contrainte structurelle supplémentaire en utilisant des dictionnaires formés d'une union de codes à permutation normalisés, de type II selon lequel toutes les permutations et tous les signes sont autorisés. Pour chaque code à permutation de type II, on appelle leader absolu, le plus grand vecteur, au sens lexicographique, que l'on obtient en ordonnant les valeurs absolues des composantes dans l'ordre décroissant. La construction des dictionnaires revient à déterminer leurs leaders absolus normalisés. Appliquer l'extension contrôlée à ces leaders absolus consiste alors à leur insérer des 0 en dernières composantes.

On se fixe en outre un critère de distorsion. Préférentiellement, le critère de distorsion choisi est ici la distance euclidienne. Le dictionnaire étant normalisé, la recherche du vecteur-code qui minimise la distance euclidienne avec un vecteur d'entrée à quantifier revient à rechercher le vecteur-code qui maximise le

produit scalaire avec ce vecteur d'entrée. De plus, le dictionnaire étant l'union de codes à permutation, la recherche du vecteur-code maximisant le produit scalaire avec un vecteur d'entrée revient à rechercher parmi les leaders absolus du dictionnaire celui qui maximise le produit scalaire avec le leader absolu de ce vecteur d'entrée (lequel est aussi obtenu par permutation des valeurs absolues de ses composantes pour les arranger dans l'ordre décroissant).

On définit ci-après une séquence d'apprentissage pour la conception des quantificateurs vectoriels au sens de l'invention. Comme indiqué ci-avant, il est préférable de déterminer une séquence d'apprentissage pour la conception d'un quantificateur. Une longue séquence constituée de trames de 289 coefficients MDCT normalisés par le facteur d'échelle de leur bande est d'abord obtenue à partir de nombreux échantillons de signaux audio en bande élargie. Puis, pour chaque vecteur normalisé de coefficients, on déduit son leader absolu. A partir de l'ensemble de leaders absolus de différentes dimensions, on crée deux catégories de séquences d'apprentissage multidimensionnelles S^0 et S^1 :

- $S^0 = \{S_j^0\}_{j \in [1,15]}$, S_j^0 étant l'ensemble de tous les vecteurs formés par les j premières composantes des leaders absolus ayant j coefficients non nuls. S_j^0 est constitué ainsi par les leaders absolus de dimension j n'ayant aucun coefficient nul, ceux de dimension j+1 ayant un seul coefficient nul, ceux de dimension j+2 ayant deux

coefficients nuls, ... ceux de dimension 15 ayant 15-j coefficients nuls,

- et $S^1 = \{S_j^1\} j \in [3,4,5,7,8,9,10,12,13,14,15]$, S_j^1 étant l'ensemble de tous les leaders absolus des bandes ayant j coefficients.

Par exemple, à partir du vecteur normalisé de coefficients (0.,0.6,0., 0.,0.8), on déduit son leader absolu (0.8, 0.6,0.,0., 0.) qui appartient à la séquence S_5^1 et un élément de S^0_2, (0.8,0.6), formé par les deux premières composantes non nulles de son leader absolu.

La première catégorie de séquences est préférentiellement utilisée pour déterminer les dictionnaires initiaux de leaders des $\overline{D'^j_{N_j}}$. La deuxième catégorie est préférentiellement utilisée pour construire des dictionnaires multidimensionnels et multirésolutions possédant les deux propriétés structurantes.

A partir de la première catégorie S^0 de séquences, on obtient un premier dictionnaire de leaders absolus normalisés pour chaque dimension j (j allant de 1 à 15) par application à la séquence S_j^0 d'un algorithme classique tel que celui dit "*des k-moyennes*". Ces leaders à composantes réelles positives sont modifiés en annulant les composantes inférieures à un seuil prédéterminé, par rapport à la première composante (c'est-à-dire la plus grande composante). Cette procédure dite de "*center-*

clippling" permet avantageusement de faire apparaître des zéros et d'extraire des leaders absolus sans composantes nulles de dimension inférieure. Pour favoriser davantage l'extension contrôlée, une transformation des composantes de ces leaders extraits est appliquée. On utilise à cet effet une normalisation de chaque leader par sa plus petite composante non nulle suivie d'une quantification scalaire uniforme de pas 1 et à niveaux de reconstruction entiers (ce qui revient à arrondir les composantes de chaque leader à l'entier le plus proche). Cette transformation amène en outre une réduction significative de la mémoire car les leaders absolus peuvent ainsi être stockés sous forme d'entiers moyennant l'introduction d'un facteur correctif de normalisation dans le calcul de distance. On notera que des leaders réels différents obtenus ou non à partir de différentes séquences S_j^0 peuvent être transformés en un même leader entier. On prévoit alors une procédure pour éliminer des redondances éventuelles et former l'ensemble $L'^0 = \bigcup_{j\in[1,\cdots 15]} L'^0_j$ de tous les leaders absolus à composantes entières non nulles, L'^0_j étant le sous-ensemble constitué par ces leaders de dimension j. Cette technique de construction de L'^0 s'inspire de la technique de construction de dictionnaires par composition partielle par extension contrôlée selon les dimensions décroissantes. On note aussi que le choix de l'ensemble A effectué a priori pourrait être revu a posteriori pour y ajouter l'élément "1" car tous les leaders de L'^0 ont au moins un "1" en dernière composante.

L'ensemble L'^0 sert de base à la composition des dictionnaires initiaux de leaders pour la conception des quantificateurs vectoriels à dimensions et résolutions multiples possédant les deux propriétés structurantes d'imbrication PR et de composition partielle par extension contrôlée PD. A partir de la séquence S^1, l'algorithme pour construire ces quantificateurs procède par dimension croissante et résolution décroissante.

Pour une dimension j, le dictionnaire initial de leaders L'^1_j est formé par tous les leaders de L'^0_j et par tous les leaders obtenus par extension contrôlée des leaders des dimensions inférieures j'(j'<j) en insérant (j-j') zéros aux leaders des ensembles $L'^0_{j'}$. Par exemple en dimension 3, on compose un dictionnaire de leaders par extension contrôlée à partir de $L'^0_1 = \{(1)\}, L'^0_2 = \{(1\,1),(2\,1),(3\,1),(4\,1),(5\,1),(9\,1)\}$, complété par les leaders de L'^0_3.

Pour chaque dimension j, l'union des codes à permutation caractérisée par L'^1_j constitue un dictionnaire de haute résolution, éventuellement supérieure à la résolution maximale désirée. Ces codes à permutation effectuent donc une partition naturelle de ce dictionnaire, chaque classe de cette partition étant un code à permutation représenté par son leader. La construction des régions du plus proche voisin correspondant aux classes de cette partition est alors effectuée par quantification de la séquence S^1. La partition est ordonnée selon le cardinal croissant des codes à permutations. En cas d'égalité des cardinaux des

codes à permutation, les codes des leaders obtenus par extension contrôlée sont favorisés par rapport à ceux des leaders de L'^0_j comme indiqué ci-avant. En cas d'égalité de cardinaux de deux classes appartenant au même ensemble (soit à $D'^j_{N_j}$, soit à $\overline{D'^j_{N_j}}$), les classes sont ordonnées selon un critère combinant le cardinal de leur région de quantification et leur contribution à la décroissance de la distorsion totale. Le cumul des cardinalités des codes à permutation ainsi ordonnés est calculé pour chaque code à permutation ainsi que le débit correspondant par vecteur. On note L^1_j l'ensemble des leaders de L'^1_j ainsi ordonné. Pour éviter une procédure de mise en train binaire conjointe des indices codés, on choisit de n'utiliser que des résolutions entières.

Les dictionnaires multi-résolutions imbriqués en résolution, en référence au tableau de la figure 7c, sont donc constitués en choisissant comme dernier code à permutation de chaque résolution différente celui dont le débit du cumul des cardinaux est le plus proche de l'entier immédiatement supérieur. Si la résolution du dictionnaire caractérisé par L^1_j est supérieure à la résolution maximale désirée, on élimine les derniers codes à permutation inutilisés. On note $L_j\left(\subseteq L^1_j\right)$ l'ensemble final ordonné des leaders de $D^j_{N_j-1}$. A la fin des itérations sur les dimensions, si certains leaders de L'^0 ne sont pas utilisés pour composer des leaders de

$\{L_j\}j \in \{3,4,5,7,8,9,10,12,13,14,15\}$, l'ensemble L'^{0} est mis à jour en les éliminant. On note cet ensemble $L^0 = \bigcup_{j\in[1,\cdots 15]} L_j^0$.

Les tableaux des figures 7c à 7e montrent les gains en mémoire apportés par la propriété d'imbrication et par la propriété de composition partielle par extension contrôlée. Le tableau de la figure 7c compare des quantificateurs vectoriels à multiples résolutions pour différentes dimensions : les premiers quantificateurs simplement structurés en unions de codes à permutations, et les seconds quantificateurs possédant de plus la propriété d'imbrication en résolutions.

Sur la figure 7c, on note :

- j : la dimension,
- N_j : le nombre de résolutions en dimension j,
- $L_{D_i^j}$: le nombre de leaders du dictionnaire D_i^j,
- L_{D^j} : le nombre de leaders du dictionnaire $D_{N_j-1}^j$,
- $j\sum_{i=1}^{N_j-1} L_{D_i^j}$: la mémoire (en nombre de mots) nécessaire pour stocker les leaders de tous les dictionnaires en dimension j sans la propriété d'imbrication,
- jL_{D^j} : la mémoire nécessaire pour stocker les leaders de tous les dictionnaires en dimension j avec la propriété d'imbrication.

Le tableau de la figure 7d compare ces derniers

quantificateurs, utilisés pour des dimensions multiples, avec des quantificateurs possédant aussi la propriété structurante de composition partielle par extension contrôlée.

Sur la figure 7d, on note :

- j : la dimension
- L_{D^j} : le nombre de leaders du dictionnaire $D^j_{N_j-1}$,
- $\sum_{k=1}^{j} L_{D^k}$: la somme des nombres de leaders des dictionnaires de résolution maximale des dimensions 1 à j,
- $\sum_{k=1}^{j} kL_{D^k}$: la mémoire nécessaire pour stocker ces leaders sans la propriété de composition partielle par extension contrôlée,
- L_j : le nombre de leaders de l'ensemble L^0_j,
- $\sum_{k=1}^{j} L_k$: leur somme des dimensions 1 à j,
- $\sum_{k=1}^{j} kL_k$: la mémoire nécessaire pour stocker les leaders de tous les dictionnaires des dimensions 1 à j avec la propriété de composition partielle par extension contrôlée.

Le tableau de la figure 7e compare des quantificateurs vectoriels à multiples résolutions et dimensions: les premiers quantificateurs simplement structurés en union de

codes à permutations et les seconds possédant de plus les propriétés structurantes d'imbrication en résolutions et de composition partielle par extension contrôlée.

Sur la figure 7e, on note:

- j: la dimension
- N_j : le nombre de résolutions en dimension j
- $\sum_{i=1}^{N_j} L_{D_i^j}$: le nombre de leaders en dimension j à stocker pour les N_j résolutions sans la propriété d'imbrication ni la propriété d'extension partielle contrôlée
- $j\sum_{i=1}^{N_j} L_{D_i^j}$: la mémoire (en nombre de mots) nécessaire pour stocker ces leaders de tous les dictionnaires en dimension j sans ces deux propriétés
- $\sum_{k=1}^{j} k\sum_{i=1}^{N_k} L_{D_i^k}$: la mémoire (en nombre de mots) nécessaire pour stocker les leaders de tous les dictionnaires des dimensions 1 à j sans ces deux propriétés

L_j: le nombre de leaders de l'ensemble $L^0{}_j$

- $\sum_{k=1}^{j} L_k$: leur somme des dimensions 1 à j
- $\sum_{k=1}^{j} kL_k$: la mémoire nécessaire pour stocker les leaders de tous les dictionnaires des dimensions 1 à j avec les deux propriétés d'imbrication et de composition partielle par extension contrôlée.

Dans les trois tableaux, la dernière colonne montre l'importance du facteur de réduction en mémoire. La seule propriété d'imbrication permet de réduire la mémoire d'un facteur supérieur à 3 en dimension 3, 5 en dimension 7, 7 en dimension 15. Grâce à la propriété d'imbrication, au lieu de stocker tous les Leaders des D_i^j pour l'ensemble des résolutions en dimension j, on ne stocke que les leaders de $D_{N_j-1}^j$ (les leaders de L_j). L'ajout de la composition partielle par extension contrôlée permet de réduire encore la mémoire comme le montre la dernière colonne du tableau de la figure 7d. Le gain supplémentaire apporté par cette propriété est d'un facteur supérieur à :

- 1,5 en dimension 4,
- 3 en dimension 8,
- et 7 en dimension 15.

Comme le montre la figure 7e, par rapport à des quantificateurs simplement structurés en union de codes à permutations, l'emploi de quantificateurs possédant de plus les deux propriétés structurantes d'imbrication en résolutions et de composition partielle par extension contrôlée permet de réduire la mémoire d'un facteur 4 en dimension 3, 13 en dimension 7 et d'un facteur supérieur à 35 pour les dimensions supérieures à 11.

Avec la propriété de composition partielle par extension contrôlée, seuls les leaders de L^0 doivent être stockés, les leaders des $\{L_j\}$ étant retrouvés à partir d'une table de correspondance des index des leaders de L_j vers les index des leaders de L^0.

On décrit maintenant comment mettre en œuvre effectivement les quantificateurs vectoriels.

Pour mettre en oeuvre un quantificateur vectoriel de dimension j et de résolution r_i, il faut résoudre les trois problèmes suivants :

- recherche du plus proche voisin d'un vecteur d'entrée dans D_i^j,
- recherche de l'index d'un vecteur-code de D_i^j,
- et réciproquement, recherche d'un vecteur-code de D_i^j à partir de son index.

Pour ce qui concerne l'indexation, on indique qu'il existe plusieurs manières connues d'indexer les vecteurs-codes d'un dictionnaire, union de codes à permutation de type II. La numérotation employée dans le mode de réalisation s'inspire de celle utilisée pour indexer les codes sphériques du réseau de Gosset.

Pour toute dimension j(j $\in\{3,4,5,7,8,9,10,12,13,14,15\}$), chaque vecteur-code de $D_{N_j-1}^j$ est indexé par un offset caractéristique de son code à permutation, d'un indice binaire donnant sa combinaison de signes et de son rang dans son code à permutation. L'offset d'un code à permutation est le cumul des cardinalités des codes à permutation le précédant dans $D_{N_j-1}^j$. Parmi les formules de numérotation des permutations, on a choisi la formule

dite de Schalkwijk.

En plus de cette numérotation classique des vecteurs-codes de $D^j_{N_j-1}$, on utilise une table de correspondance des index des leaders de L_j vers les index des leaders de L^0. Les leaders de L^0 étant stockés, on dispose ainsi d'une grande liberté d'indexation de L^0. Par exemple, on peut classer ces leaders à composantes entières non nulles par dimension croissante. A chaque index m^j d'un leader x^j de L_j est associé un index l_m d'un leader $x^{j'}$ de L^0. A partir de cet index l_m, on retrouve la dimension j' du leader $x^{j'}$ et le leader lui-même. Le leader x^j est alors retrouvé en insérant (j-j') zéros comme dernières composantes de $x^{j'}$.

Le tableau de la figure 7f donne les 23 premiers leaders de L^0. Le tableau de la figure 7g donne les leaders des codes à permutation des dictionnaires en dimension 3 en indiquant pour chaque leader x^3 quel leader $x^{j'}$ de $L^0_{j'}$ de dimension j'(j'≤j), a été étendu pour l'obtenir. Incidemment, on remarque que si j=j', alors $x^{j'} = x^3$.

Sur la figure 7f, on note :
- l : l'indice du leader dans L^0 (parmi les 516),
- j : sa dimension,
- l^j : son indice dans les leaders de L^0_j.

Sur la figure 7g, on note :
- m^3 : l'indice du leader x^3 parmi les 23 leaders de $D^3_{N_3}$,
- i : l'indice du dictionnaire de plus petite résolution

auquel le leader appartient (i.e. $x^3 \notin D_{i-1}^3$ et $x^3 \in D_i^3$),

- jr_i : le débit par vecteur de ce dictionnaire D_i^3,
- j' : la dimension du leader $x^{j'}$ de L^0 (nombre de composantes non nulles),
- l_m : l'indice de $x^{j'}$ parmi les 516 leaders de L^0.

On décrit ci-après les algorithmes de codage et de décodage, proprement dits dans le cas général et on verra plus loin le cas particulièrement avantageux où une contrainte structurelle supplémentaire (union de codes à permutation) a été ajoutée.

On indique d'abord qu'ils exploitent préférentiellement la structure des dictionnaires induite en particulier par la propriété d'extension contrôlée qui permet de réduire la complexité de l'algorithme de recherche du plus proche voisin. En particulier, on peut grouper les vecteurs-codes ayant la même règle d'insertion. Par exemple, dans le cas d'un critère de distorsion de distance euclidienne qui sera traité en détail plus loin, si L vecteurs-codes $\left\{x_l^j, l = 0,1;\cdots L-1\right\}$ de dimension j d'un dictionnaire D_i^j sont obtenus par la même règle d'insertion $R\left(n,\{(i_m,p_m)\}m=0,n-1\right)$ à partir de L vecteurs-codes x_l^{j-n} de dimension j-n d'un dictionnaire $D_{i'}^{j-n}$, le calcul des L distances des vecteurs— codes x_l^j à un vecteur d'entrée $y : Dist\left(y, x_l^j\right) = \sum_{k=0}^{j-1}\left(y_k - x_{k,l}^j\right)^2$ peut être accéléré en calculant

d'abord le terme $\sum_{m=0}^{n-1}\left(y_{p_m}-a_{i_m}\right)^2$ puis en calculant les L distances des vecteurs-codes x_l^{j-n} au vecteur y' de dimension (j-n) obtenu en enlevant à y les n composantes y_{p_m} : $Dist\left(y', x_l^{j-n}\right)=\sum_{k=0}^{j-n-1}\left(y'_k-x_{k,l}^{j-n}\right)^2$.

Comme indiqué ci-avant, pour chaque dimension, seule une partie du dictionnaire de résolution maximale doit être stockée, les autres vecteurs-codes se déduisent à partir d'éléments pris dans les dictionnaires de résolution maximale de dimension inférieure et de règles d'insertion.

On donne ci-après un exemple détaillé de réalisation du codage/décodage en compression dans l'utilisation du procédé de création de dictionnaire selon l'invention.

On indique tout d'abord qu'au lieu de stocker, pour toutes les dimensions j à considérer, l'ensemble de tous les dictionnaires $\left\{D_i^j\right\}_{i=1,..,N_j}$, on ne stocke donc que les $\{\overline{D'^j_{N_j}}\}$ ainsi que des tables de correspondance. Ces tables permettent de reconstituer un vecteur-code de $D_{N_j}^j$ à partir de son indice. Comme décrit ci-avant, il y a plusieurs façons d'élaborer ces tables et donc de les stocker. Par exemple, on peut, pour toutes les dimensions j à considérer, tabuler pour chaque indice m_j (d'un vecteur-code x^j de $D_{N_j}^j$) trois valeurs entières scalaires : j', m' et l_r , où l_r est le numéro de la règle d'insertion

qui permet de reconstituer x^j par composition partielle par extension contrôlée appliquée à l'élément d'indice m' de l'ensemble de $\overline{D'^{j'}_{N_{j'}}}$. Les tables de correspondance ne requièrent au plus que le stockage de $3\sum_{j=1}^{N} T^j_{N_j}$ mots (on rappelle que T_i^j est la taille du dictionnaire D_i^j). Quant au stockage proprement dit des dictionnaires d'un quantificateur vectoriel à résolutions et dimensions multiples, il requière $\sum_{j=1}^{N} j \sum_{i=1}^{N_j} T_i^j$ mots dans le cas d'un quantificateur vectoriel ne possédant pas les deux propriétés structurantes d'imbrication en résolutions et de composition partielle par extension, tandis que le stockage des dictionnaires d'un quantificateur vectoriel possédant ces deux propriétés structurantes ne requière lui que $\sum_{j=1}^{N} j\overline{T'^j_{N_j}}$ mots, en notant $\overline{T'^j_{N_j}}$ la taille de l'ensemble $\overline{D'^j_{N_j}}$ ($\overline{T'^j_{N_j}} \leq T^j_{N_j}$). Cependant, de manière générale, $\overline{T'^j_{N_j}}$ est beaucoup plus petit que $T^j_{N_j}$, car on cherche bien entendu à favoriser l'ensemble $D'^j_{N_j}$ par rapport à l'ensemble $\overline{D'^j_{N_j}}$. Quelques exemples numériques de gain en stockage seront donnés dans un mode de réalisation décrit plus loin.

L'algorithme de codage qui consiste à rechercher le plus proche voisin x^j dans D_i^j d'un vecteur d'entrée

$y=(y_0,\ldots,y_k,\ldots,y_{j-1})$ comporte préférentiellement les étapes suivantes :

L'étape CO0) consiste en une étape d'initialisation où l'on pose :

d_{min} = VALMAX; m_{min} = -1; m^j=0

Pour tout indice $m^j \in [0, T_i^j [$:

L'étape suivante CO1) consiste en la reconstitution du vecteur-code x^j d'indice m^j et s'effectue préférentiellement comme suit :

a) lecture des trois indices j', m' et l_r dans les tables de correspondance associées à $D_{N_j}^j$,

b) lecture dans l'ensemble $\overline{D'^{j'}_{N_{j'}}}$ du vecteur $x^{j'}$ de dimension j' et d'indice m',

c) reconstitution du vecteur-code x^j par application à $x^{j'}$ de la propriété de composition partielle par extension contrôlée selon la règle d'insertion d'indice l_r.

L'étape CO2) consiste à calculer la distance $d(y,x^j)$ entre y et x^j selon le critère de distorsion choisi.

Les étapes suivantes CO3) et CO4) consistent à répéter les opérations CO1) et CO2) pour identifier l'indice de vecteur dont la distance au vecteur d'entrée est minimale.

Ainsi :

* si $d(y,x^j) < d_{min}$ alors $d_{min} = d(y,x^j)$ et $m_{min} = m^j$
* ensuite, on incrémente m^j : $m^j=m^j+1$
* on prévoit un test de fin :

si $m^j < T_i^j$ aller à l'étape CO1),

sinon : arrêter.

A l'étape de fin CO5), on détermine le vecteur-code plus proche voisin du vecteur d'entrée $\boldsymbol{y}$ en tant que vecteur-code dont l'indice m_{min} a été identifié en correspondance de la distance la plus petite d_{min} avec le vecteur d'entrée $\boldsymbol{y}$.

Ainsi, l'algorithme se poursuit par l'étape CO5) :

* Fin

le plus proche voisin x^j de $\boldsymbol{y}$ dans D_i^j est le vecteur-code d'indice m_{min}

L'algorithme de décodage qui consiste à rechercher un vecteur-code de D_i^j à partir de son index est donné par l'étape CO1) de l'algorithme de codage. On indique, en particulier, que le décodage implique la reconstitution complète du vecteur-code x^j (opération c) de l'étape CO1)), quel que soit l'indice à décoder.

En revanche, au codage, cette reconstitution peut être partielle. En effet, elle peut parfois être omise si le critère de distorsion dans le calcul de distance de l'étape CO2) peut être décomposé en deux termes:

- un dépendant uniquement de l'indice de la règle d'insertion,
- et un autre du vecteur-code $x^{j'}$.

Par exemple, dans le cas d'un critère de distorsion de distance euclidienne, on peut, à l'étape d'initialisation CO0), pré-calculer, pour chaque règle d'insertion d'indice l_r utilisée dans D_i^j, la distance $d_{l_r} = \sum_{m=0}^{j-j'-1} \left(y_{p_m} - a_{i_m}\right)^2$ (si la règle d'insertion d'indice l_r consiste à insérer j-j' composantes a_{i_m} en positions p_m , m allant de 0 à j-j'-1). Le calcul de la distance entre $\boldsymbol{y}$ et le vecteur $x^j(j', m', l_r)$ de l'étape CO2) revient alors à calculer la distance : $d(y', x^{j'}) = \sum_{k=0}^{j'-1} \left(y'_k - x_k^{j'}\right)^2$, où :

- $x^{j'}$ est le vecteur obtenu à l'opération b) de l'étape CO1),
- et y' le vecteur de dimension j', obtenu en enlevant à y les j-j' composantes y_{p_m},
- la distance $d(y, x^j)$ étant alors obtenue par simple sommation $d(y, x^j) = d_{l_r} + d(y', x^{j'})$.

C'est la raison pour laquelle on a défini, ci-avant, de "*partielle*" la reconstruction d'un vecteur-code $x^{j'}$ de dimension j' inférieure à la dimension j (qui serait la dimension d'un vecteur-code x^j complètement reconstruit), pendant le processus de codage.

D'autre part, si un vecteur $x^{j'}$ intervient plusieurs fois dans la composition de vecteurs-codes de D_i^j (avec différentes règles d'insertion), on peut aussi pré-calculer à l'étape d'initialisation, les termes $d(y', x^{j'})$. On voit donc que le compromis stockage

(temporaire)/complexité du codage peut être ajusté selon le besoin de l'application.

De même, le compromis stockage/complexité d'indexation peut aussi être ajusté au besoin de l'application.

Pour le codage, dans le cas de la contrainte supplémentaire d'une union de codes à permutations précitée, l'algorithme de recherche du plus proche voisin, pour les codes sphériques du réseau régulier de Gosset en dimension 8, se généralise aisément en se simplifiant à ces dictionnaires, par union de codes à permutation de type II.

Un tel algorithme de recherche est décrit en particulier dans :
- "*Algorithme de Quantification Vectorielle Algébrique Sphérique par le Réseau de Gosset* E_8", C.Lamblin, J.P.Adoul, Annales Des Télécommunications, n° 3-4, 1988.

Une première simplification est apportée par la "*liberté*" des signes des codes à permutation de type II que ne possèdent pas les codes à permutation du réseau de Gosset à composantes impaires. Une deuxième simplification est apportée par la prise en compte du nombre de composantes non nulles de chaque leader pour le calcul du produit scalaire. Ceci illustre l'exploitation de la structure induite par la propriété de composition partielle par extension contrôlée par l'algorithme de codage. Une dernière modification tient compte du stockage sous forme entière des leaders de L^0, ce qui conduit à introduire

dans le calcul du produit scalaire un facteur correctif égal à l'inverse de la norme euclidienne de ces leaders à composantes entières strictement positives.

On décrit ci-après une réalisation dans laquelle la recherche du plus proche voisin d'un vecteur d'entrée y de dimension j dans le dictionnaire D_i^j exploite, en plus des deux propriétés structurantes de l'invention, la structure en union de codes à permutation précitée.

On prévoit globalement trois étapes supplémentaires :

- deux étapes préliminaires (avant l'étape de reconstruction CO1) ci-avant) pour déterminer le leader absolu $|\tilde{y}|$ et le vecteur signe ε du vecteur à coder (étapes CP1) et CP2)),
- et une dernière étape pour calculer le rang de son plus proche voisin dans le dictionnaire (étape CP5)).

La recherche décrite ci-avant est effectuée, non plus parmi les T_i^j vecteurs-codes de D_i^j (i.e. non plus pour $m^j \in [0, T_i^j[$), mais seulement sur l'ensemble $L_j(i)$ des $L_{D_i^j}$ leaders de D_i^j (pour $m^j \in [0, L_{D_i^j}[$, en notant $L_{D_i^j}$ le nombre de leaders ou de codes à permutations de D_i^j).

Dans ce mode de réalisation, la recherche du plus proche voisin de y dans D_i^j revient à rechercher d'abord le plus

proche voisin de $|\tilde{y}|$ dans l'ensemble $L_j(i)$ (parmi les $L_{D_i^j}$ premiers leaders de L_j). Comme décrit ci-avant, il n'est pas nécessaire de reconstituer complètement ces leaders (opération c) de l'étape CO1)), le critère de distorsion (ici le produit scalaire modifié) n'étant calculé que sur les composantes non nulles de chaque leader. Il suffit donc de déterminer pour chaque leader, le leader correspondant dans L^0 utilisant la table de correspondance des index des leaders de L_j vers les index des leaders de L^0 associant à chaque index m^j d'un leader x^j de L_j un index l_m d'un leader $x^{j'}$ de L^0.

L'algorithme se déroule alors préférentiellement selon l'exemple qui suit :

*Etape CP1):
Passage du vecteur d'entrée $y=(y_0,\ldots,y_k,\ldots,y_{j-1})$ à son vecteur absolu $|y|=(|y_0|,\ldots,|y_k|,\ldots,|y_{j-1}|)$ et à son vecteur signe $\varepsilon=(\varepsilon_0,\ldots,\varepsilon_k,\ldots,\varepsilon_{j-1})$ avec $\varepsilon_k=1$ si $y_k\geq 0$ et $\varepsilon_k=-1$ sinon.

*Etape CP2):
Recherche du leader $|\tilde{y}|$ de $|y|$ par permutation de ses composantes pour les arranger dans l'ordre décroissant
*Etape CP3) :
*Etape Co0'): Initialisation :
ps_{max} = -1.; m_{max} = -1; m^j=0

pour tout indice $m^j \in [0, L_{D_i^j}[$

*Etape CO1'): reconstruction du leader d'indice m^j :

a) Lecture de l'indice l_m du leader $x^{j'}$ associé au leader d'indice m^j de L_j , dans la table de correspondance associant les leaders de L_j à ceux de L^0, puis détermination de la dimension j' du leader $x^{j'}$ et lecture du facteur correctif α (avec $\frac{1}{\alpha} = \sqrt{\sum_{k=0}^{j'-1} \left(x_k^{j'}\right)^2}$)

b) Lecture dans l'ensemble L^0 du leader $x^{j'}$ de dimension j' et d'indice l_m.

*Etape CO2') Calcul du produit scalaire modifié entre $|\tilde{y}|$ et $x^{j'}$: $ps(|\tilde{y}|, x^{j'}) = \alpha \sum_{k=0}^{j'-1} \left(|\tilde{y}_k| \cdot x_k^{j'}\right)$

Les étapes suivantes consistent à répéter les opérations CO1') et CO2') pour identifier l'indice du leader-code dont le produit scalaire modifié avec le leader absolu du vecteur d'entré est maximal. Ainsi :

si $ps(|\tilde{y}|, x^{j'}) > ps_{max}$ alors $ps_{max} = ps(|\tilde{y}|, x^{j'})$ et $m_{max} = m^j$

* ensuite, on incrémente m^j: $m^j = m^j + 1$

* Test de fin

si $m^j < L_{D_i^j}$ aller à l'étape CO1'), sinon arrêter,

A cette étape de fin, on calcule l'index du plus proche voisin de y dans D_i^j par la procédure d'indexation d'une union de codes à permutation à partir du numéro du code à permutation m_{max} trouvé à l'étape CP3), du rang de la permutation effectuée à l'étape CP2) et du vecteur de signes déterminé à l'étape CP1).

Il faut noter que l'étape CP2) peut être accélérée. En effet, si n_i^j est le nombre maximum de composantes non nulles des leaders de $L_j(i)$, il suffit de rechercher les n_i^j plus grandes composantes de $|y|$. Il existe plusieurs variantes de l'étape CP3) selon le compromis stockage/complexité désiré. Si l'on veut minimiser le nombre de calculs, on peut tabuler pour tous les leaders de L^0 simplement leur dimension j' et leur facteur correctif. La détermination de la dimension j' mentionnée à l'étape CP3) consiste dans ce cas en une lecture de la table de correspondance. A l'inverse, si l'on veut plutôt réduire la mémoire, cette détermination est calculée à partir de l'indice l_m. De même, le facteur correctif peut être calculé après la lecture du leader $x^{j'}$.

Ainsi, l'algorithme de recherche du plus proche voisin d'un vecteur d'entrée y de dimension j dans le dictionnaire D_i^j, utilisant une structure en union de codes à permutation, peut se résumer préférentiellement comme suit :

CP1) on passe du vecteur d'entrée $y = (y_0,\ldots,y_k,\ldots,y_{j-1})$ à son vecteur absolu $|y| = (|y_0|,\cdots,|y_k|,\cdots,|y_{j-1}|)$ et à son vecteur signe $\varepsilon = (\varepsilon_0,\ldots,\varepsilon_k,\ldots,\varepsilon_{j-1})$ avec $\varepsilon_k=1$ si $y_k \geq 0$ et $\varepsilon_k = -1$ sinon,

CP2) on recherche le leader $|\tilde{y}|$ de $|y|$ par permutation de ses composantes pour les arranger dans l'ordre décroissant,

CP3) on recherche le plus proche voisin de $|\tilde{y}|$ dans l'ensemble $L_j(i)$ des leaders de D_i^j (en fait parmi les M_i^j premiers leaders de L_j en notant M_i^j le nombre de codes à permutations de D_i^j). Comme indiqué ci-avant, cette étape revient à rechercher le leader de L^0 qui maximise le produit scalaire modifié parmi la liste des M_i^j leaders de L^0 indiquée par la table de correspondance des index des leaders de L_j vers les index des leaders de L^0. Si la dimension d'un leader $x^{j'}$ de L^0 est j' ($j' \leq j$), le calcul de son produit scalaire avec $|\tilde{y}|$ n'est effectué que sur les j' premières composantes de $|\tilde{y}|$, puis multiplié par l'inverse de la norme euclidienne de $x^{j'}$.

CP4) et on calcule l'index du rang de ce plus proche voisin de y dans D_i^j par la procédure d'indexation d'une union de codes à permutation à partir du numéro du code à permutation trouvé à l'étape précédente, du rang de la permutation effectuée à l'étape CP2) et du vecteur de signes déterminé à l'étape CP1).

En bref, l'étape CP2) peut être accélérée. En effet, si n_i^j est le nombre maximum de composantes non nulles des leaders de $L_j(i)$, il suffit de rechercher les n_i^j plus grandes composantes de $|y|$.

On décrit maintenant un algorithme de décodage, au sens général, sans nécessairement utiliser limitativement une indexation d'union de codes à permutation décrite ci-avant en tant que réalisation avantageuse. L'algorithme de décodage se présente préférentiellement comme suit.

A partir d'un index m_j reçu, on détermine si cet indice correspond à un vecteur-code appartenant à $\overline{D'^{j}_{N_j-1}}$ ou à $D'^{j}_{N_j-1}$.

Dans le premier cas, m_j est associé à un index unique dans $\overline{D'^{j}_{N_j-1}}$, et le vecteur-code est obtenu par une simple lecture de table de correspondance.

Dans le deuxième cas, m_j pointe sur un élément $\overline{D'^{j'}_{N_{j'}-1}}$ ($j'<j$) et sur une règle d'insertion.

La détermination de l'appartenance de $x^j_{m_j}$ à $D'^{j}_{N_j-1}$ ou à son complémentaire peut être effectuée de différentes façons. Par exemple, on peut utiliser une indication binaire pour chaque indice. On peut aussi pour chaque résolution $\mathbf{r}_i$ indexer les éléments du complémentaire D^j_{i-1} dans D^j_i en commençant par les éléments obtenus par extension contrôlée appartenant à D'^j_i, suivis des éléments "*libres*" appartenant à $\overline{D'^j_i}$. L'appartenance à $D'^{j}_{N_j-1}$ ou à $\overline{D'^{j}_{N_j-1}}$ est alors faite par de simples tests. De même, la règle d'insertion peut être explicitement indexée ou non.

Par exemple, dans les modes de réalisation décrits ci-après, la règle d'insertion est implicitement retrouvée à partir de l'index. On comprendra aussi que le compromis stockage/complexité d'indexation peut être ajusté en fonction des besoins de l'application.

On revient ici au cas particulier de la contrainte supplémentaire définie par l'union de codes à permutation.

Préférentiellement, l'algorithme de décodage s'inspire du document :

- "*Algorithme de Quantification Vectorielle Algébrique Sphérique par le Réseau de Gosset* E_8", C. Lamblin, J.P. Adoul, Annales Des Télécommunications, n° 3-4, 1988,

en utilisant en plus la table de correspondance des indices de leaders de L_j vers ceux de L^0.

A partir de l'index d'un vecteur-code dans D_i^j, on détermine l'indice de son leader dans $L_j(i)$, son rang dans son code à permutation et le signe de ses composantes non nulles. La table de correspondance donne alors l'index du leader dans L^0 qui alors est obtenu par une simple lecture de table stockée en mémoire ainsi que de son facteur de normalisation qui permet de normaliser le vecteur-code décodé.

Un autre exemple de mise en oeuvre de la présente invention est donné ci-après. Cet exemple repose encore sur le codeur par transformée de type TDAC, mais pour une utilisation pour coder des signaux audio numériques

échantillonnés à 32 kHz et de 15 kHz de largeur de bande (bande FM), contrairement à l'exemple donné ci-avant de l'utilisation du codeur TDAC en bande élargie pour coder des signaux audio numériques échantillonnés à 16 kHz.

Le principe de ce codeur est similaire à celui du codeur TDAC en bande élargie à 16 kHz. Le signal audio, limité en bande à 16 kHz et échantillonné maintenant à 32 kHz, est aussi découpé en trames de 20 ms. Ceci conduit après transformation MDCT à obtenir 640 coefficients. Le spectre est découpé en 52 bandes de largeurs inégales, la découpe de la bande élargie étant identique à la découpe effectuée par le codeur TDAC en bande élargie.

Le tableau de la figure 8a donne la découpe en bandes utilisée et la dimension résultante du vecteur des coefficients (correspondant au nombre de coefficients indiqué à la troisième colonne).

La quantification de l'enveloppe spectrale utilise aussi un codage d'Huffman et le débit variable restant est alloué dynamiquement aux coefficients à partir de la version déquantifiée de cette enveloppe spectrale.

La quantification des coefficients MDCT utilise des dictionnaires construits selon l'invention. Comme dans le cas décrit précédemment, les dictionnaires sont aussi structurés en union de codes à permutation. Pour les dimensions inférieures à 15, les quantificateurs vectoriels sont les mêmes que ceux de la bande élargie. On construit ainsi des dictionnaires pour les dimensions 16,

17,18, 19, 20 et 24. Pour la dimension 24, on a en outre combiné cette structure à la structure en produit cartésien. La dernière bande haute de 24 coefficients est découpée en deux vecteurs de dimension 12 : l'un est formé par les coefficients pairs, l'autre par les coefficients impairs. Ici, les quantificateurs vectoriels construits pour la dimension 12 ont été exploités.

Le tableau de la figure 8b donne le nombre de résolutions différentes ainsi que leurs valeurs pour les dimensions 1 à 24.

La présente invention fournit ainsi une solution efficace au problème de la quantification vectorielle à débit et à dimension variables. L'invention résout conjointement les deux problèmes de résolution et de dimension variables en prévoyant un quantificateur vectoriel dont les dictionnaires, pour les différentes dimensions et résolutions, possèdent les propriétés structurantes PR et PD ci-avant.

Pour une dimension donnée, l'imbrication des dictionnaires garantit, d'une part, la décroissance locale de la distorsion en fonction de la résolution et réduit, d'autre part, notablement la quantité de mémoire requise pour le stockage car les dictionnaires des résolutions inférieures n'ont pas à être stockés, tous les éléments de ces dictionnaires étant en effet dans le dictionnaire de résolution maximale. Par rapport au quantificateur vectoriel structuré en arbre des figures 1a et 1b, le choix d'imbriquer les dictionnaires apporte donc déjà deux

avantages: l'assurance d'une décroissance de la distorsion locale en fonction des résolutions croissantes et un stockage réduit. Elle permet aussi une grande finesse de résolution avec, si nécessaire, une granularité inférieure au bit, facilitant le choix de dictionnaires de tailles non obligatoirement égales à des puissances de 2. Cette granularité fine des résolutions est particulièrement intéressante si plusieurs vecteurs de dimension et/ou de résolution variables sont à quantifier par trame, en associant à ces quantificateurs de débit par vecteur non entier un algorithme de mise en train binaire des indices.

La propriété d'imbrication PR des dictionnaires permet de n'avoir à stocker que les dictionnaires de résolution maximale. Grâce à la deuxième propriété PD, la quantité de mémoire de stockage est encore plus réduite. En effet, une partie des éléments des dictionnaires de résolution maximale n'a pas à être stockée car elle se déduit à partir d'éléments pris dans les dictionnaires de résolution maximale mais de dimension inférieure, en tenant compte de règles d'insertion $\{R_m\}$ prédéfinies. La proportion d'éléments ainsi structurés est aisément adaptable et permet d'ajuster finement la quantité de mémoire de stockage.

La structure induite par ces deux propriétés PR et PD permet donc de réduire avantageusement la mémoire nécessaire de stockage. Elle peut évidemment l'être davantage en imposant aux dictionnaires des contraintes structurelles supplémentaires telles que celles déjà mentionnées dans la partie introductive en référence à

l'état de l'art ci-avant. Dans des modes de réalisation préférés, on prévoit par exemple l'utilisation de quantificateurs vectoriels sphériques, union de codes à permutation, combinée le cas échéant à la structure en produit cartésien décrite ci-avant.

Par rapport aux quantificateurs vectoriels algébriques, cette structure de dictionnaires induite par les deux propriétés offre une très grande souplesse de conception tant pour le choix des dimensions que pour celui des résolutions. De plus, ces quantificateurs vectoriels s'adaptent à la statistique de la source à coder et évitent ainsi le problème de la conception délicate d'un "*companding vectoriel*" obligatoire en quantification vectorielle algébrique pour rendre uniforme la distribution de la source à coder.

REVENDICATIONS

1. Dictionnaire comportant des vecteurs-codes de dimension variable et destiné à être utilisé dans un dispositif de codage et/ou décodage en compression de signaux numériques, par quantification vectorielle à débit variable définissant une résolution variable,

caractérisé en ce qu'il comporte :

- d'une part, pour une dimension donnée, des dictionnaires de résolution croissante imbriqués les uns dans les autres,
- et, d'autre part, pour une dimension donnée, une union :
  - d'un premier ensemble constitué de vecteurs-codes construits en insérant, dans des vecteurs-codes de dictionnaires de dimension inférieure, des éléments pris dans un ensemble fini de nombres réels selon un jeu fini de règles d'insertion prédéterminées,
  - et d'un deuxième ensemble constitué de vecteurs-codes ne pouvant être obtenus par insertion dans des vecteurs-codes de dimension inférieure des éléments dudit ensemble fini selon ledit jeu de règles d'insertion.

2. Dictionnaire selon la revendication 1, caractérisé en ce que ledit jeu de règles d'insertion est élaboré à partir de règles élémentaires consistant à insérer un seul élément de l'ensemble fini de réels en tant que composante à une position donnée d'un vecteur.

3. Dictionnaire selon la revendication 2, caractérisé en ce que chaque règle élémentaire est définie par un couple

de deux entiers positifs représentatifs :
- d'un rang de l'élément dans ledit ensemble fini,
- et d'une position d'insertion.

4. Procédé pour former un dictionnaire selon l'une des revendications 1 à 3, le dictionnaire comportant des vecteurs-codes de dimension variable et destiné à être utilisé dans un dispositif de codage et/ou décodage en compression de signaux numériques, par quantification vectorielle à débit variable définissant une résolution variable,
dans lequel, pour une dimension donnée :
a) on construit un premier ensemble constitué de vecteurs-codes formés en insérant/supprimant dans des vecteurs-codes de dictionnaires de dimension inférieure/supérieure des éléments pris dans un ensemble fini de nombres réels selon un jeu fini de règles d'insertion/suppression prédéterminées,
b) on construit, pour ladite dimension donnée, un premier dictionnaire, intermédiaire, comportant au moins ledit premier ensemble,
c) et, pour adapter ledit dictionnaire à une utilisation avec au moins une résolution donnée, on construit, à partir du dictionnaire intermédiaire, un second dictionnaire, définitif, par imbrication/simplification de dictionnaires de résolutions croissantes/décroissantes, les dictionnaires de résolutions croissantes étant imbriqués les uns dans les autres du dictionnaire de plus petite résolution jusqu'au dictionnaire de plus grande résolution.

5. Procédé selon la revendication 4, dans lequel, pour une dimension donnée **N** :

a0) on obtient un dictionnaire initial de dimension initiale **n**, inférieure à ladite dimension donnée **N**,

a1) on construit un premier ensemble constitué de vecteurs-codes de dimension **n+i** formés en insérant dans des vecteurs-codes du dictionnaire initial des éléments pris dans un ensemble fini de nombres réels selon un jeu fini de règles d'insertion prédéterminées,

a2) on prévoit un deuxième ensemble constitué de vecteurs-codes de dimension **n+i** ne pouvant être obtenus par insertion dans les vecteurs-codes du dictionnaire initial des éléments dudit ensemble fini avec ledit jeu de règles d'insertion,

a3) on construit un dictionnaire intermédiaire, de dimension **n+i** comportant une union dudit premier ensemble et dudit second ensemble,

et on répète, au plus **N-n-1** fois, les étapes a1) à a3), avec ledit dictionnaire intermédiaire en tant que dictionnaire initial, jusqu'à ladite dimension donnée **N**.

6. Procédé selon la revendication 4, dans lequel, pour une dimension donnée **N** :

a'0) on obtient un dictionnaire initial de dimension initiale **n**, supérieure à ladite dimension donnée **N**,

a'1) on construit un premier ensemble, de dimension **n-i**, par sélection et extraction de vecteurs-codes possibles de dimension **n-i** dans le dictionnaire de dimension **n**, selon un jeu fini de règles de suppression prédéterminées,

a'2) on prévoit un deuxième ensemble constitué de vecteurs-codes de dimension **n-i**, ne pouvant être obtenus

par suppression, dans les vecteurs-codes du dictionnaire initial, des éléments dudit ensemble fini avec ledit jeu de règles de suppression,

a'3) on construit un dictionnaire intermédiaire, de dimension **n-i** comportant une union dudit premier ensemble et dudit second ensemble,

et on répète, au plus **n-N-1** fois, les étapes a'1) à a'3), avec ledit dictionnaire intermédiaire en tant que dictionnaire initial, jusqu'à ladite dimension donnée **N**.

7. Procédé selon les revendications 5 et 6, dans lequel on obtient **N** dictionnaires successifs de dimensions respectives **1** à **N**, à partir d'un dictionnaire initial de dimension **n**, par la mise en œuvre répétée des étapes a1) à a3) pour les dimensions **n+1** à **N**, et par la mise en œuvre répétée des étapes a'1) à a'3) pour les dimensions **n-1** à **1**.

8. Procédé selon l'une des revendications 4 à 7, dans lequel ledit jeu de règles d'insertion/suppression est élaboré à partir de règles élémentaires consistant à insérer/supprimer un seul élément de l'ensemble fini de réels en tant que composante à une position donnée d'un vecteur.

9. Procédé selon la revendication 8, dans lequel chaque règle élémentaire est définie par un couple de deux entiers positifs représentatifs :

- d'un rang de l'élément dans ledit ensemble fini,
- et d'une position d'insertion/suppression.

10. Procédé selon l'une des revendications 4 à 9, dans lequel on définit a priori ledit ensemble fini et ledit jeu de règles d'insertion/suppression, avant de construire le dictionnaire par analyse d'une source à quantifier.

11. Procédé selon la revendication 10, dans lequel ladite source est modélisée par une séquence d'apprentissage et la définition dudit ensemble fini et dudit jeu de règles d'insertion/suppression est effectuée par analyse statistique de ladite source.

12. Procédé selon l'une des revendications 10 et 11, dans lequel ledit ensemble fini est choisi par estimation d'une densité de probabilité monodimensionnelle de ladite source.

13. Procédé selon l'une des revendications 4 à 9, dans lequel on définit a posteriori ledit ensemble fini et ledit jeu de règles d'insertion/suppression, après construction de dictionnaires par imbrication/simplification de dictionnaires de résolutions successives, suivie d'une analyse statistique de ces dictionnaires ainsi construits.

14. Procédé selon les revendications 10 et 13, dans lequel :
- on choisit, a priori, un premier ensemble et un premier jeu de règles d'insertion/suppression par analyse d'une séquence d'apprentissage, pour former un ou plusieurs dictionnaires intermédiaires,
- on met à jour au moins une partie dudit premier ensemble

et/ou dudit premier jeu de règles d'insertion/suppression, par analyse a posteriori desdits un ou plusieurs dictionnaires intermédiaires,
- et, le cas échéant, on met à jour aussi au moins une partie de l'ensemble des vecteur-codes formant lesdits un ou plusieurs dictionnaires intermédiaires.

15. Procédé selon l'une des revendications 4 à 14, dans lequel l'étape c) comporte les opérations suivantes :
c0) on obtient un dictionnaire initial de résolution initiale r_n, inférieure à ladite résolution donnée r_N,
c1) à partir du dictionnaire initial, on construit un dictionnaire intermédiaire de résolution r_{n+1} supérieure à la résolution initiale r_n,
c2) on répète l'opération c1) jusqu'à atteindre la résolution donnée r_N.

16. Procédé selon la revendication 15, dans lequel, pour chaque itération de l'opération c1), on prévoit une construction de classes et de centroïdes dans laquelle les centroïdes appartenant au moins aux dictionnaires de résolution supérieure à une résolution courante r_i sont recalculés et mis à jour.

17. Procédé selon la revendication 16, dans lequel les centroïdes qui appartiennent aux dictionnaires de résolution inférieure à une résolution courante r_i ne sont mis à jour que si les distorsions totales de tous les dictionnaires de résolution inférieure sont décroissantes d'une mise à jour à l'autre.

18. Procédé selon l'une des revendications 4 à 14, dans lequel l'étape c) comporte les opérations suivantes :
c'0) on obtient un dictionnaire initial de résolution initiale r_n, supérieure à ladite résolution donnée r_N,
c'1) à partir du dictionnaire initial, on construit un dictionnaire intermédiaire de résolution r_{n-1} inférieure à la résolution initiale r_n, par partition du dictionnaire initial en plusieurs sous-ensembles ordonnés selon un critère prédéterminé, et
c'2) on répète l'opération c'1) jusqu'à atteindre la résolution donnée r_N.

19. Procédé selon la revendication 18, dans lequel ledit critère prédéterminé est choisi parmi le cardinal des sous-ensembles, une sollicitation des sous-ensembles dans une séquence d'apprentissage, une contribution des sous-ensembles à une distorsion totale ou préférentiellement à une décroissance de cette distorsion.

20. Procédé selon l'une des revendications 18 et 19, dans lequel ladite partition utilise une partie au moins desdites règles d'insertion/suppression.

21. Procédé selon les revendications 15 et 18, dans lequel on obtient **N** dictionnaires successifs de résolutions respectives r_1 à r_N, à partir d'un dictionnaire initial de résolution intermédiaire r_n, par la mise en œuvre répétée de l'étape c1) pour les résolutions croissantes r_{n+1} à r_N, et par la mise en œuvre répétée de l'étape c'1) pour les résolutions décroissantes r_{n-1} à r_1.

22. Procédé selon l'une des revendications 4 à 21, dans lequel, pour adapter ledit dictionnaire à une utilisation avec une dimension donnée **N** de vecteurs-codes, on inverse sensiblement les étapes a) et b), d'une part, et l'étape c), d'autre part, de sorte que :

- à l'étape c), on construit, à partir d'un dictionnaire initial de résolution r_n et de dimension **N'**, un premier dictionnaire, intermédiaire, toujours de dimension **N'** mais de résolution r_N supérieure/inférieure, par imbrication/simplification de dictionnaires de résolutions croissantes/décroissantes, pour atteindre sensiblement la résolution r_N dudit premier dictionnaire,
- à l'étape a), pour atteindre la dimension donnée **N**, on construit un premier ensemble constitué de vecteurs-codes formés en insérant/supprimant, dans des vecteurs-codes du premier dictionnaire de dimension **N'** inférieure/supérieure à ladite dimension donnée **N**, des éléments pris dans un ensemble fini de nombres réels selon un jeu fini de règles d'insertion/suppression prédéterminées,
- et, à l'étape b), suite à une étape éventuelle d'adaptation définitive à la résolution r_N, on construit, pour ladite dimension donnée **N**, un second dictionnaire, définitif, comportant au moins ledit premier ensemble.

23. Procédé selon l'une des revendications 4 à 22, dans lequel on stocke dans une mémoire, une fois pour toutes, ledit jeu de règles d'insertion/suppression, identifiées chacune par un indice (l_r), et, pour une dimension donnée :

- ledit deuxième ensemble constitué de vecteurs-codes ne pouvant être obtenus par application de

l'insertion/suppression à des vecteurs-codes de dimension inférieure/supérieure à la dimension donnée selon ledit jeu de règles d'insertion/suppression,
- ainsi qu'au moins une table de correspondance permettant de reconstituer un vecteur-code quelconque du dictionnaire de dimension donnée, en utilisant les indices des règles d'insertion/suppression et des indices identifiant des éléments dudit second ensemble,
ce qui permet d'éviter le stockage complet du dictionnaire pour ladite dimension donnée, en stockant simplement les éléments dudit second ensemble et des liens dans la table de correspondance pour accéder à ces éléments et aux règles d'insertion/suppression associées.

24. Procédé selon la revendication 23, dans lequel les tables de correspondances sont élaborées au préalable, pour chaque indice (m^j) d'un vecteur-code (x^j) du dictionnaire ($D^j_{N_j}$) de dimension donnée (j) pouvant être reconstruit à partir d'éléments d'indices courants (m') dans le second ensemble de dimension courante (j'), par une tabulation de trois valeurs scalaires entières représentant :
- une dimension courante (j') dudit second ensemble,
- un indice courant (m') d'un élément du second ensemble, et
- un indice (l_r) de règle d'insertion/suppression,
cette règle d'insertion/suppression au moins contribuant à reconstituer ledit vecteur-code (x_j) du dictionnaire (D^j_{Nj}) de dimension donnée (j), en appliquant l'insertion/suppression à l'élément dudit indice courant (m') et de ladite dimension courante (j').

25. Utilisation du dictionnaire obtenu par la mise en œuvre du procédé selon l'une des revendications 23 et 24, au codage/décodage en compression de signaux numériques, par quantification vectorielle à débit variable définissant une résolution variable, dans laquelle on recherche le vecteur-code (x^j) le plus proche voisin d'un vecteur d'entrée $y=(y_0,\ldots,y_k,\ldots,y_{j-1})$ dans un dictionnaire (D^i_j) de dimension donnée (j),
et comprenant les étapes suivantes :
CO1) pour un indice courant (m^j) dudit vecteur-code (x^j) recherché, reconstitution au moins partielle d'un vecteur-code d'indice (m') correspondant audit indice courant (m^j), au moins par lecture préalable des indices (j',m',l_r) figurant dans les tables de correspondance permettant d'élaborer ledit dictionnaire,
CO2) au moins au codage, calcul d'une distance entre le vecteur d'entrée et le vecteur-code reconstitué à l'étape CO1),
CO3) au moins au codage, répétition des étapes CO1) et CO2), pour tous les indices courants dans ledit dictionnaire,
CO4) au moins au codage, identification de l'indice (m_{min}) du vecteur-code au moins partiellement reconstitué dont la distance (d_{min}) avec le vecteur d'entrée, calculée au cours de l'une des itérations de l'étape CO2), est la plus petite, et
CO5) au moins au décodage, détermination du plus proche voisin du vecteur d'entrée (y) en tant que vecteur-code (x^j) dont l'indice (m_{min}) a été identifié à l'étape CO4).

26. Utilisation selon la revendication 25, dans laquelle l'étape CO1), au moins au décodage, comporte :

CO11) la lecture, dans les tables de correspondance, d'indices représentatifs de liens vers ledit second ensemble et vers les règles d'insertion/suppression et incluant :

- l'indice d'une dimension courante d'un sous-ensemble dudit second ensemble,
- l'indice courant d'un élément dudit sous-ensemble,
- et l'indice de la règle d'insertion/suppression appropriée pour la construction du vecteur-code du dictionnaire de dimension donnée, à partir dudit élément,

CO12) la lecture, dans le sous-ensemble identifié par sa dimension courante, dudit élément identifié par son indice courant,

CO13) la reconstitution complète du vecteur-code à ladite dimension donnée en appliquant audit élément lu à l'étape CO12) la règle d'insertion/suppression appropriée et identifiée par son indice lu à l'étape CO11).

27. Utilisation selon la revendication 25, dans laquelle, au codage,

* l'étape CO1) comporte :

CO11) la lecture, dans les tables de correspondance, d'indices représentatifs de liens vers ledit second ensemble et vers les règles d'insertion/suppression et incluant :

- l'indice d'une dimension courante d'un sous-ensemble dudit second ensemble,
- l'indice courant d'un élément dudit sous-ensemble,

- et l'indice de la règle d'insertion/suppression appropriée pour la construction du vecteur-code du dictionnaire de dimension donnée,

CO12) la lecture, dans le sous-ensemble identifié par sa dimension courante, dudit élément identifié par son indice courant,

* à l'étape CO2), on calcule ladite distance en fonction d'un critère de distorsion estimé en fonction de :

- l'indice de la règle d'insertion/suppression,
- et de l'élément du sous-ensemble identifié par son indice courant,

ce qui permet de ne construire que partiellement le vecteur-code à ladite dimension donnée à l'étape CO1), en réservant la reconstruction complète simplement au décodage.

28. Utilisation selon l'une des revendications 25 à 27, dans laquelle on prévoit en outre une propriété structurante supplémentaire selon une union de codes à permutation et exploitant une indexation de ladite union de codes à permutation, et dans laquelle :

CP1) à partir d'un signal d'entrée, on forme un vecteur d'entrée $y=(y_0, \ldots, y_k, \ldots, y_{j-1})$ défini par son vecteur absolu $|y|=(|y_0|,\cdots,|y_k|,\cdots,|y_{j-1}|)$ et par un vecteur signe $\varepsilon=(\varepsilon_0,\ldots,\varepsilon_k,\ldots,\varepsilon_{j-1})$ avec $\varepsilon_k=\pm 1$,

CP2) on range les composantes du vecteur $|y|$ par valeurs décroissantes, par permutation, pour obtenir un vecteur leader $|\tilde{y}|$,

CP3) on détermine, parmi les leaders du dictionnaire D^j_i de dimension j, un plus proche voisin $x^{j'}$ du vecteur

leader $|\tilde{y}|$,

CP4) on détermine un index du rang dudit plus proche voisin $x^{j'}$ dans le dictionnaire D^j_i,

CP5) et l'on applique une valeur effective de codage/décodage au vecteur d'entrée, qui est fonction dudit index déterminé à l'étape CP4), de la ladite permutation déterminée à l'étape CP2) et dudit vecteur signe déterminé à l'étape CP1).

29. Utilisation selon l'une des revendications 25 à 28, dans laquelle on stocke au moins lesdites tables de correspondance dans une mémoire d'un dispositif de codage/décodage.

30. Produit programme d'ordinateur destiné à être stocké dans une mémoire d'une unité de traitement, notamment d'un ordinateur ou d'un terminal mobile, ou sur un support mémoire amovible et destiné à coopérer avec un lecteur de l'unité de traitement,
caractérisé en ce qu'il comporte des instructions pour la mise en œuvre du procédé selon l'une des revendications 4 à 24.

31. Produit programme d'ordinateur destiné à être stocké dans une mémoire d'une unité de traitement, notamment d'un ordinateur ou d'un terminal mobile intégrant un dispositif de codage/décodage, ou sur un support mémoire amovible et destiné à coopérer avec un lecteur de l'unité de traitement,
caractérisé en ce qu'il comporte des instructions pour la mise en œuvre de l'application au codage/décodage en

compression selon l'une des revendications 25 à 29.

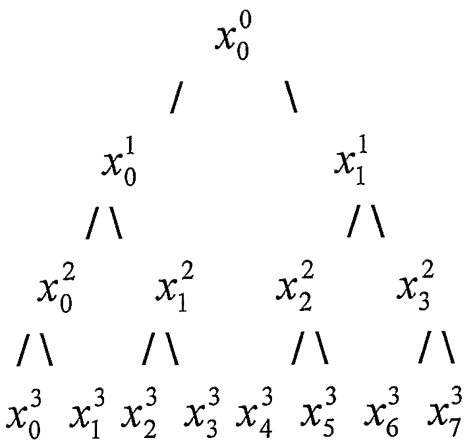


FIG. 1a

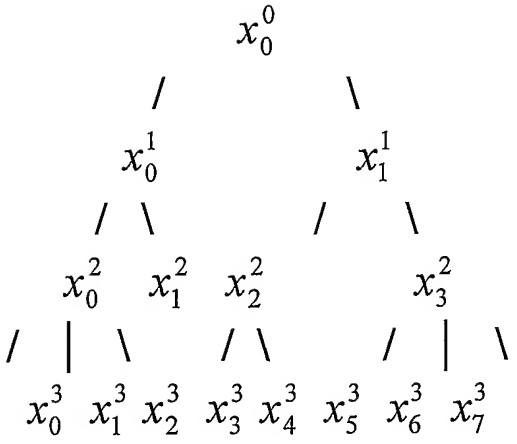


FIG. 1b

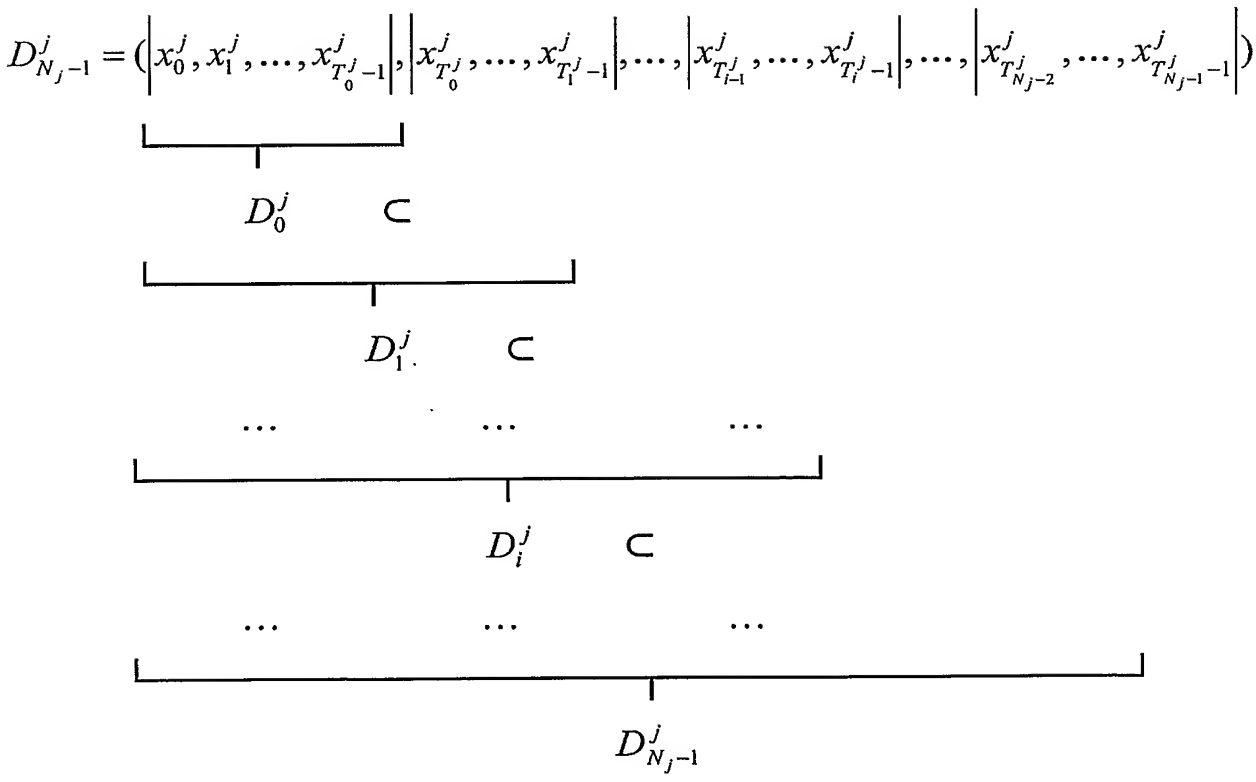


FIG. 3

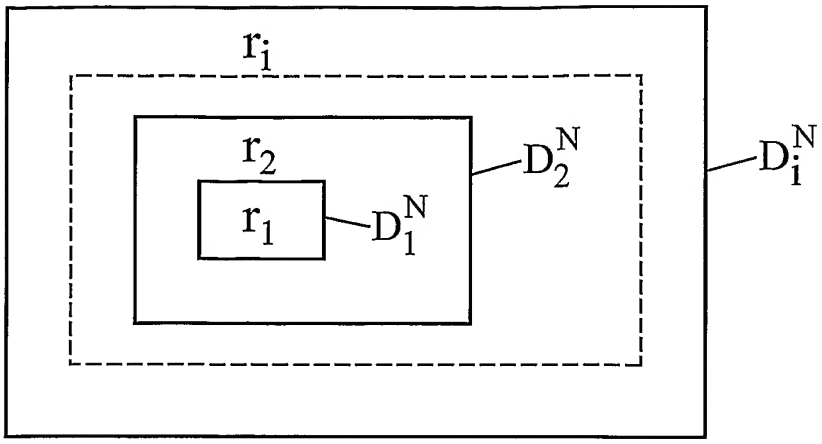

r_i
r_2
r_1
D_1^N
D_2^N
D_i^N

FIG. 2a

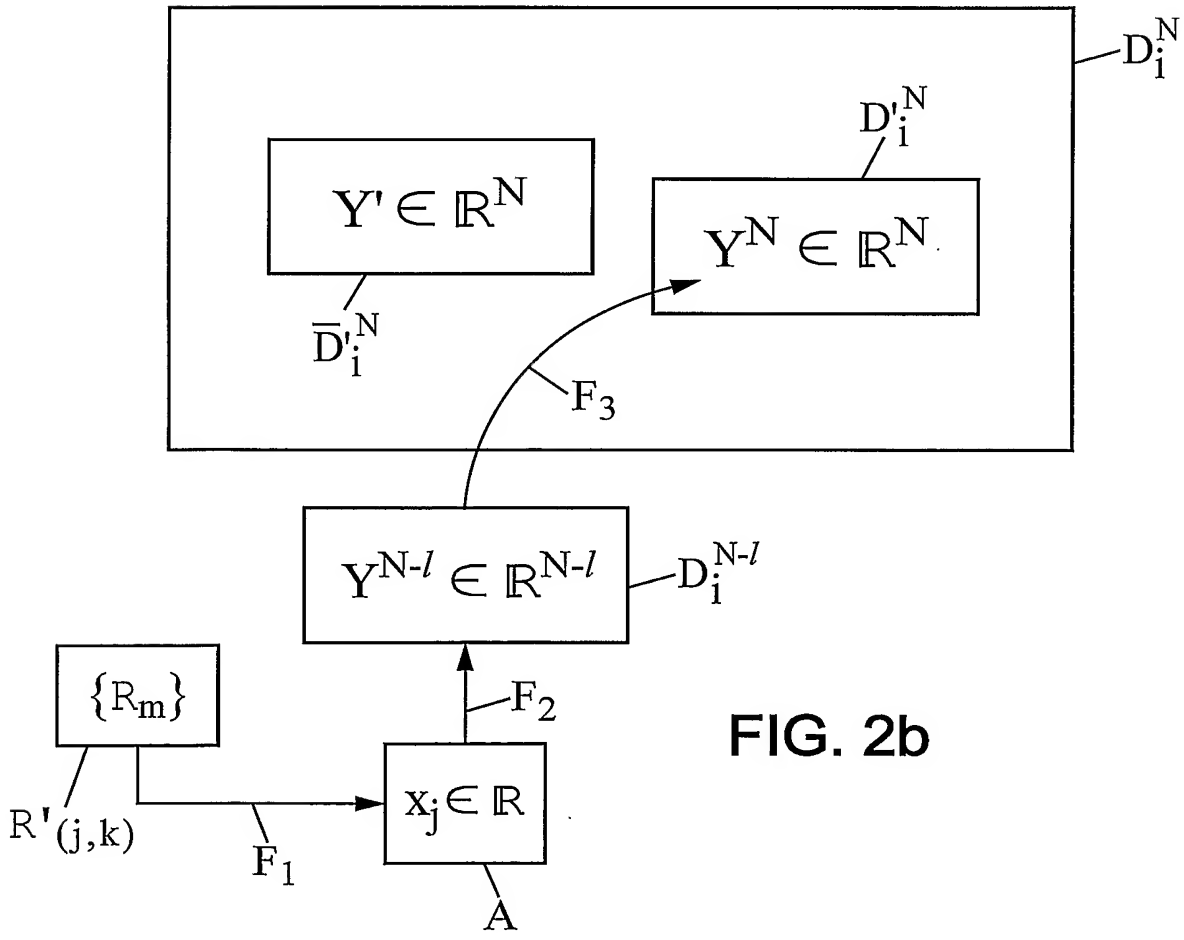

D_i^N
D'^N_i
$Y' \in \mathbb{R}^N$
$Y^N \in \mathbb{R}^N$
$\overline{D}'^N_i$
F_3
$Y^{N-l} \in \mathbb{R}^{N-l}$
D_i^{N-l}
$\{R_m\}$
F_2
$R'(j,k)$
$x_j \in \mathbb{R}$
F_1
A

FIG. 2b

$x^j=(x_0^j,\cdots,x_k^j,\cdots,x_{j-1}^j)\rightarrow$
×
$\rightarrow x^{j+1}=(x_0^j,a_0,x_1^j,\cdots,x_k^j,\cdots,x_{j-1}^j)$
$R(1,\{(0,1)\}=R'(0,1)$
$x^j=(x_0^j,\cdots,x_k^j,\cdots,x_{j-1}^j)\rightarrow$
×
$\rightarrow x^{j+2}=(a_1,x_0^j,a_0,x_1^j,\cdots,x_k^j,\cdots,x_{j-1}^j)$
$R(2,\{(1,0),(0,2)\}=R'(1,0)\circ R'(0,2)$
$x^j=(x_0^j \cdots x_k^j \cdots x_{j-1}^j)\rightarrow$
×
$\rightarrow x^{j+n}=(x_0^j, \cdots x_{p_0-1}^j, a_{i_0}, x_{p_0}^j, \cdots x_{p_1-2}^j, a_{i_1}, x_{p_1-1}^j, \cdots x_{p_m-m-1}^j a_{i_m} x_{p_m-m}^j \cdots x_{p_{n-1}-n}^j a_{i_{n-1}} x_{p_{n-1}-n+1}^j \cdots x_{j-1}^j)$
$0 \cdots k \cdots j-1 \quad k \quad 0 \cdots p_0-1 \; p_0 \; p_0+1 \cdots p_1-1 \; p_1 \; p_1+1 \cdots p_m-1 \; p_m \; p_m+1 \cdots p_{n-1}-1 \; p_{n-1} \; p_{n-1}+1 \cdots j+n-1$
$R(n,\{(i_m,p_m)\}_{m=0,n-1})=\circ_{m=0,n-1} R'(i_m,p_m),\ p_0<p_1\ldots<p_m\ldots<p_{n-1}$

FIG. 4

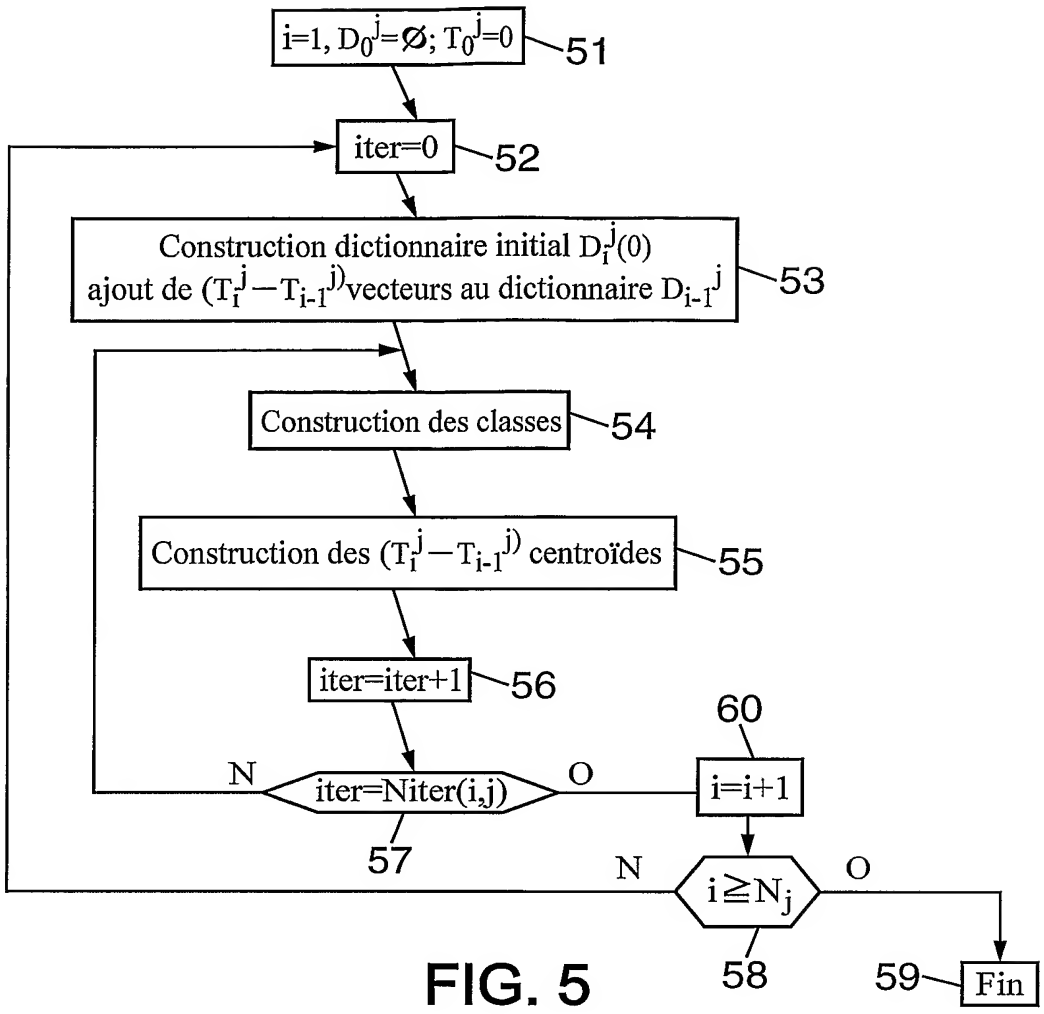

i=1, $D_0^j=\varnothing$; $T_0^j=0$
51
iter=0
52
Construction dictionnaire initial $D_i^j(0)$
ajout de $(T_i^j - T_{i-1}^j)$ vecteurs au dictionnaire D_{i-1}^j
53
Construction des classes
54
Construction des $(T_i^j - T_{i-1}^j)$ centroïdes
55
iter=iter+1
56
N
iter=Niter(i,j)
O
57
60
i=i+1
N
$i \geqq N_j$
O
58
59
Fin

FIG. 5

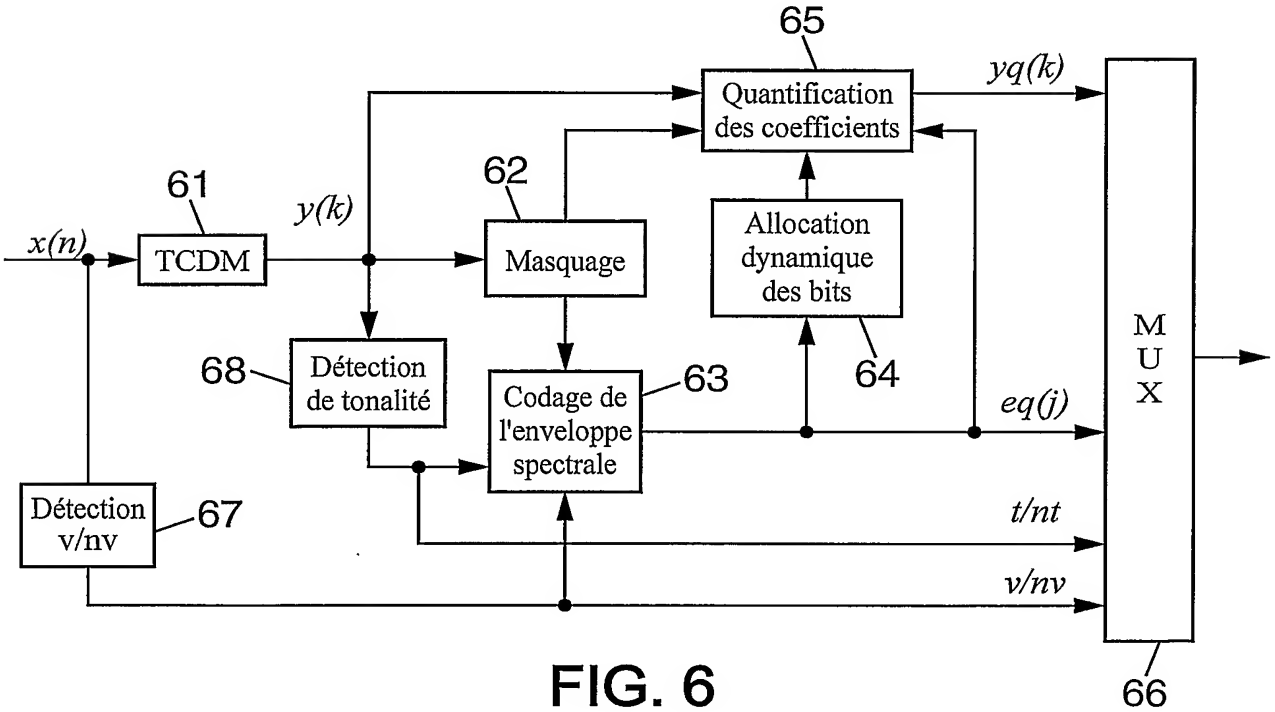

65
Quantification des coefficients
yq(k)
61
x(n)
TCDM
y(k)
62
Masquage
Allocation dynamique des bits
64
68
Détection de tonalité
Codage de l'enveloppe spectrale
63
eq(j)
M U X
Détection v/nv
67
t/nt
v/nv
66

FIG. 6

Bande	Borne Supérieure (Hz)	Nb. de coefficients	Bande	Borne Supérieure (Hz)	Nb. De coefficients
0	75	3	16	2375	10
1	150	3	17	2625	10
2	225	3	18	2875	10
3	300	3	19	3175	12
4	375	3	20	3475	12
5	475	4	21	3775	12
6	575	4	22	4075	12
7	675	4	23	4400	13
8	800	5	24	4725	13
9	925	5	25	5050	13
10	1050	5	26	5400	14
11	1225	7	27	5750	14
12	1425	8	28	6100	14
13	1650	9	29	6475	15
14	1875	9	30	6850	15
15	2125	10	31	7225	15

FIG. 7a

FIG. 7b

j	N_j	Débit par bande $\left\{jr_0^j, jr_1^j, \ldots, jr_i^j, \ldots, jr_{N_j-1}^j\right\}$
1	1	1
2	5	2, 3, 4, 5, 6
3	8	3, 4, 5, 6, 7, 8, 9,10
4	10	3, 5, 6, 7, 8, 9,10,11,12,13
5	13	4, 6, 7, 8, 9,10,11,12,13,14,15,16,17
6	15	4, 7, 8, 9,10,11,12,13,14,15,16,17,18,19,20
7	18	4, 7, 8, 9,10,11,12,13,14,15,16,17,18,19,20,21,22,23
8	20	4, 7, 9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26
9	23	5, 8, 9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26,27,28,29
10	25	5, 8,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26,27,28,29,30,31,32
11	25	5, 8,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26,27,28,29,30,31,32
12	25	5, 9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26,27,28,29,30,31,32
13	25	5, 9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26,27,28,29,30,31,32
14	24	5, 9,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26,27,28,29,30,31,32
15	24	5, 9,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26,27,28,29,30,31,32

FIG. 7e

j	N_j	$\sum_{i=1}^{N_j} L_{D_i^j}$	$j\sum_{i=1}^{N_j} L_{D_i^j}$	$\sum_{k=1}^{j} k\sum_{i=1}^{N_k} L_{D_i^k}$	L_j	$\sum_{k=1}^{j} kL_k$	$\sum_{k=1}^{j} kL_k / \sum_{k=1}^{j} k\sum_{i=1}^{N_k} L_{D_i^k}$
1	1	1	1	1	1	1	1
2	5	18	36	37	6	13	0,351
3	8	71	213	250	16	61	0,244
4	10	166	664	914	27	169	0,185
5	13	418	2090	3004	48	409	0,136
6	15	695	4170	7174	48	697	0,097
7	18	1095	7665	14839	63	1138	0,077
8	20	1620	12960	27799	72	1714	0,062
9	23	2251	20259	48058	73	2371	0,049
10	25	3057	30570	78628	83	3201	0,041
11	25	2447	26917	105545	10	3311	0,031
12	25	2101	25212	130757	33	3707	0,028
13	25	1826	23738	154495	19	3954	0,026
14	24	1608	22512	177007	10	4094	0,023
15	24	1447	21705	198712	7	4199	0,021

j	N_j	$\sum_{i=1}^{N_j-1} L_{D_i^j}$	$j\sum_{i=1}^{N_j-1} L_{D_i^j}$	L_{D^j}	jL_{D^j}	$jL_{D^j} / j\sum_{i=1}^{N_j-1} L_{D_i^j}$
1	1	1	1	1	1	1.00
2	5	18	36	7	14	0.39
3	8	71	213	23	69	0.32
4	10	166	664	50	200	0.30
5	13	418	2090	98	490	0.23
6	15	695	4170	146	876	0.21
7	18	1095	7665	209	1463	0.19
8	20	1620	12960	281	2248	0.17
9	23	2251	20259	354	3186	0.16
10	25	3057	30570	437	4370	0.14
11	25	2447	26917	369	4059	0.15
12	25	2101	25212	321	3852	0.15
13	25	1826	23738	268	3484	0.15
14	24	1608	22512	233	3262	0.14
15	24	1447	21705	205	3075	0.14

FIG. 7c

j	L_{D^j}	$\sum_{k=1}^{j} L_{D^k}$	$\sum_{k=1}^{j} kL_{D^k}$	L_j	$\sum_{k=1}^{j} L_k$	$\sum_{k=1}^{j} kL_k$	$\sum_{k=1}^{j} kL_k / \sum_{k=1}^{j} kL_{D^k}$
1	1	1	1	1	1	1	1.00
2	7	8	15	6	7	13	0.87
3	23	31	84	16	23	61	0.73
4	50	81	284	27	50	169	0.60
5	98	179	774	48	98	409	0.53
6	146	325	1650	48	146	697	0.42
7	209	534	3113	63	209	1138	0.37
8	281	815	5361	72	281	1714	0.32
9	354	1169	8547	73	354	2371	0.28
10	437	1606	12917	83	437	3201	0.25
11	369	1975	16976	10	447	3311	0.20
12	321	2296	20828	33	480	3707	0.18
13	268	2564	24312	19	499	3954	0.16
14	233	2797	27574	10	509	4094	0.15
15	205	3002	30649	7	516	4199	0.14

FIG. 7d

l	j	l^j	Leader	l	j	l^j	Leader
0	1	0	1	12	3	5	4 1 1
1	2	0	1 1	13	3	6	4 4 1
2	2	1	2 1	14	3	7	5 1 1
3	2	2	3 1	15	3	8	6 1 1
4	2	3	4 1	16	3	9	3 2 1
5	2	4	5 1	17	3	10	4 2 1
6	2	5	9 1	18	3	11	4 3 1
7	3	0	1 1 1	19	3	12	5 2 1
8	3	1	2 1 1	20	3	13	5 4 1
9	3	2	2 2 1	21	3	14	7 3 1
10	3	3	3 1 1	22	3	15	8 2 1
11	3	4	3 3 1				

FIG. 7f

m^3	i	jr_i	leader x^3	j'	$x^{j'}$	l_m	m^3	i	jr_i	leader x^3	j'	$x^{j'}$	l_m
0	0	3	1 0 0	1	1	0	12	6	9	4 1 1	3	4 1 1	12
1	1	4	1 1 1	3	1 1 1	7	13	6	9	4 4 1	3	4 4 1	13
2	2	5	1 1 0	2	1 1	1	14	6	9	5 1 1	3	5 1 1	14
3	3	6	3 1 0	2	3 1	3	15	6	9	6 1 1	3	6 1 1	15
4	4	7	4 1 0	2	4 1	4	16	6	9	3 2 1	3	3 2 1	16
5	4	7	5 1 0	2	5 1	5	17	6	9	4 2 1	3	4 2 1	17
6	4	7	9 1 0	2	9 1	6	18	6	9	4 3 1	3	4 3 1	18
7	5	8	2 1 0	2	2 1	2	19	7	10	5 2 1	3	5 2 1	19
8	5	8	2 1 1	3	2 1 1	8	20	7	10	5 4 1	3	5 4 1	20
9	5	8	2 2 1	3	2 2 1	9	21	7	10	7 3 1	3	7 3 1	21
10	5	8	3 1 1	3	3 1 1	10	22	7	10	8 2 1	3	8 2 1	22
11	5	8	3 3 1	3	3 3 1	11							

FIG. 7g

Bande	Borne Supérieure (Hz)	Nb. de coefficients	Bande	Borne Supérieure (Hz)	Nb. de coefficients	Bande	Borne Supérieure (Hz)	Nb. de coefficients
0	75	3	18	2875	10	35	8725	15
1	150	3	19	3175	12	36	9100	15
2	225	3	20	3475	12	37	9500	16
3	300	3	21	3775	12	38	9900	16
4	375	3	22	4075	12	39	10300	16
5	475	4	23	4400	13	40	10700	16
6	575	4	24	4725	13	41	11125	17
7	675	4	25	5050	13	42	11575	18
8	800	5	26	5400	14	43	12025	18
9	925	5	27	5750	14	44	12475	18
10	1050	5	28	6100	14	45	12925	18
11	1225	7	29	6475	15	46	13400	19
12	1425	8	30	6850	15	47	13900	20
13	1650	9	31	7225	15	48	14400	20
14	1875	9	32	7600	15	49	14900	20
15	2125	10	33	7975	15	50	15400	20
16	2375	10	34	8350	15	51	16000	24
17	2625	10						

FIG. 8a

j	N_j	$\left\{jr_0^j, jr_1^j, \ldots, jr_i^j, \ldots, jr_{N_j-1}^j\right\}$
1	1	1
2	5	2, 3, 4, 5, 6
3	8	3, 4, 5, 6, 7, 8, 9,10
4	10	3, 5, 6, 7, 8, 9,10,11,12,13
5	13	4, 6, 7, 8, 9,10,11,12,13,14,15,16,17
6	15	4, 7, 8, 9,10,11,12,13,14,15,16,17,18,19,20
7	18	4, 7, 8, 9,10,11,12,13,14,15,16,17,18,19,20,21,22,23
8	20	4, 7, 9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26
9	23	5, 8, 9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26,27,28,29
10	25	5, 8,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26,27,28,29,30,31,32
11	25	5, 8,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26,27,28,29,30,31,32
12	25	5, 9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26,27,28,29,30,31,32
13	25	5, 9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26,27,28,29,30,31,32
14	24	5, 9,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26,27,28,29,30,31,32
15	24	5, 9,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26,27,28,29,30,31,32
16	23	9,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26,27,28,29,30,31,31
17	23	10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26,27,28,29,30,31,31
18	23	10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26,27,28,29,30,31,31
19	23	10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26,27,28,29,30,31,31
20	23	10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26,27,28,29,30,31,31
21	0	$\varnothing$
22	0	$\varnothing$
23	0	$\varnothing$
24	24	9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26,27,28,29,30,31,31

FIG. 8b

INTERNATIONAL SEARCH REPORT

[redacted]national Application No
PCT/FR2004/000219

A. CLASSIFICATION OF SUBJECT MATTER
IPC 7 H03M7/30

According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)
IPC 7 H03M

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

Electronic data base consulted during the international search (name of data base and, where practical, search terms used)
EPO-Internal, INSPEC, WPI Data

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category °	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
Y	COMBESCURE P ET AL: "A 16, 24, 32 kbit/s wideband speech codec based on ATCELP" 1999 IEEE INTERNATIONAL CONFERENCE ON ACOUSTICS, SPEECH, AND SIGNAL PROCESSING, 15 March 1999 (1999-03-15), pages 5-8, XP010327999 PHOENIX, AZ, USA ISBN: 0-7803-5041-3	1-3
A	page 6, right-hand column, line 41 - page 7, left-hand column, line 2 figure 3a	4-31

-/--

[X] Further documents are listed in the continuation of box C.

[X] Patent family members are listed in annex.

° Special categories of cited documents :

"A" document defining the general state of the art which is not considered to be of particular relevance

"E" earlier document but published on or after the international filing date

"L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)

"O" document referring to an oral disclosure, use, exhibition or other means

"P" document published prior to the international filing date but later than the priority date claimed

"T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention

"X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone

"Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art.

"&" document member of the same patent family

Date of the actual completion of the international search	Date of mailing of the international search report
11 October 2004	12/11/2004

Name and mailing address of the ISA	Authorized officer
European Patent Office, P.B. 5818 Patentlaan 2 NL - 2280 HV Rijswijk Tel. (+31-70) 340-2040, Tx. 31 651 epo nl, Fax: (+31-70) 340-3016	Winkler, G

Form PCT/ISA/210 (second sheet) (January 2004)

INTERNATIONAL SEARCH REPORT

[redacted]national Application No
PCT/FR2004/000219

C.(Continuation) DOCUMENTS CONSIDERED TO BE RELEVANT

Category °	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
Y	MINJIE XIE ET AL: "Embedded algebraic vector quantizers (EAVQ) with application to wideband speech coding" 1996 IEEE INTERNATIONAL CONFERENCE ON ACOUSTICS, SPEECH, AND SIGNAL PROCESSING, vol. 1, 7 May 1996 (1996-05-07), pages 240-243, XP002252600 NEW YORK, IEEE, US ISBN: 0-7803-3193-1	1-3
A	page 241, left-hand column, line 17 - right-hand column, line 42	4-31
A	CONWAY J. H., SLOANE N. J. A.: "Fast Quantizing and Decoding Algorithms for lattice Quantizers and Codes" IEEE TRANSACTIONS ON INFORMATION THEORY, vol. 28, no. 2, March 1982 (1982-03), pages 227-232, XP002300007 NEW YORK, IEEE, US page 230, right-hand column, line 6 - line 32	1-31
A	WO 03/103151 A (VOICEAGE CORP ; BESSETTE BRUNO (CA); RAGOT STEPHANE (CA); ADOUL JEAN-P) 11 December 2003 (2003-12-11) page 1 - page 14	1-31
A	LAMBLIN C ET AL: "ALGORITHME DE QUANTIFICATION VECTORIELLE SPHERIQUE A PARTIR DU RESEAU DE GOSSET D'ORDRE 8 SPHERICAL VECTOR QUANTIZATION ALGORITHM BASED ON AN EIGHT DIMENSIONAL GOSSET LATTICE" ANNALES DES TELECOMMUNICATIONS, LES MOULINEAUX, FR, vol. 43, no. 3/4, 1 March 1988 (1988-03-01), pages 172-186, XP000572495	1-31

INTERNATIONAL SEARCH REPORT

Information on patent family members

[redacted]national Application No
PCT/FR2004/000219

Patent document cited in search report	Publication date	Patent family member(s)	Publication date
WO 03103151 A	11-12-2003	CA 2388358 A1	30-11-2003
		WO 03103151 A1	11-12-2003

RAPPORT DE RECHERCHE INTERNATIONALE

[redacted]ande Internationale No
PCT/FR2004/000219

A. CLASSEMENT DE L'OBJET DE LA DEMANDE
CIB 7 H03M7/30

Selon la classification internationale des brevets (CIB) ou à la fois selon la classification nationale et la CIB

B. DOMAINES SUR LESQUELS LA RECHERCHE A PORTE

Documentation minimale consultée (système de classification suivi des symboles de classement)
CIB 7 H03M

Documentation consultée autre que la documentation minimale dans la mesure où ces documents relèvent des domaines sur lesquels a porté la recherche

Base de données électronique consultée au cours de la recherche internationale (nom de la base de données, et si réalisable, termes de recherche utilisés)
EPO-Internal, INSPEC, WPI Data

C. DOCUMENTS CONSIDERES COMME PERTINENTS

Catégorie °	Identification des documents cités, avec, le cas échéant, l'indication des passages pertinents	no. des revendications visées
Y	COMBESCURE P ET AL: "A 16, 24, 32 kbit/s wideband speech codec based on ATCELP" 1999 IEEE INTERNATIONAL CONFERENCE ON ACOUSTICS, SPEECH, AND SIGNAL PROCESSING, 15 mars 1999 (1999-03-15), pages 5-8, XP010327999 PHOENIX, AZ, USA ISBN: 0-7803-5041-3	1-3
A	page 6, colonne de droite, ligne 41 - page 7, colonne de gauche, ligne 2 figure 3a	4-31

-/--

[X] Voir la suite du cadre C pour la fin de la liste des documents

[X] Les documents de familles de brevets sont indiqués en annexe

° Catégories spéciales de documents cités:

"A" document définissant l'état général de la technique, non considéré comme particulièrement pertinent

"E" document antérieur, mais publié à la date de dépôt international ou après cette date

"L" document pouvant jeter un doute sur une revendication de priorité ou cité pour déterminer la date de publication d'une autre citation ou pour une raison spéciale (telle qu'indiquée)

"O" document se référant à une divulgation orale, à un usage, à une exposition ou tous autres moyens

"P" document publié avant la date de dépôt international, mais postérieurement à la date de priorité revendiquée

"T" document ultérieur publié après la date de dépôt international ou la date de priorité et n'appartenenant pas à l'état de la technique pertinent, mais cité pour comprendre le principe ou la théorie constituant la base de l'invention

"X" document particulièrement pertinent; l'inven tion revendiquée ne peut être considérée comme nouvelle ou comme impliquant une activité inventive par rapport au document considéré isolément

"Y" document particulièrement pertinent; l'inven tion revendiquée ne peut être considérée comme impliquant une activité inventive lorsque le document est associé à un ou plusieurs autres documents de même nature, cette combinaison étant évidente pour une personne du métier

"&" document qui fait partie de la même famille de brevets

Date à laquelle la recherche internationale a été effectivement achevée	Date d'expédition du présent rapport de recherche internationale
11 octobre 2004	12/11/2004

Nom et adresse postale de l'administration chargée de la recherche internationale	Fonctionnaire autorisé
Office Européen des Brevets, P.B. 5818 Patentlaan 2 NL – 2280 HV Rijswijk Tel. (+31–70) 340–2040, Tx. 31 651 epo nl, Fax: (+31–70) 340–3016	Winkler, G

RAPPORT DE RECHERCHE INTERNATIONALE

[redacted]ande Internationale No
PCT/FR2004/000219

C.(suite) DOCUMENTS CONSIDERES COMME PERTINENTS

Catégorie °	Identification des documents cités, avec, le cas échéant, l'indication des passages pertinents	no. des revendications visées
Y	MINJIE XIE ET AL: "Embedded algebraic vector quantizers (EAVQ) with application to wideband speech coding" 1996 IEEE INTERNATIONAL CONFERENCE ON ACOUSTICS, SPEECH, AND SIGNAL PROCESSING, vol. 1, 7 mai 1996 (1996-05-07), pages 240-243, XP002252600 NEW YORK, IEEE, US ISBN: 0-7803-3193-1	1-3
A	page 241, colonne de gauche, ligne 17 - colonne de droite, ligne 42	4-31
A	CONWAY J. H., SLOANE N. J. A.: "Fast Quantizing and Decoding Algorithms for lattice Quantizers and Codes" IEEE TRANSACTIONS ON INFORMATION THEORY, vol. 28, no. 2, mars 1982 (1982-03), pages 227-232, XP002300007 NEW YORK, IEEE, US page 230, colonne de droite, ligne 6 - ligne 32	1-31
A	WO 03/103151 A (VOICEAGE CORP ; BESSETTE BRUNO (CA); RAGOT STEPHANE (CA); ADOUL JEAN-P) 11 décembre 2003 (2003-12-11) page 1 - page 14	1-31
A	LAMBLIN C ET AL: "ALGORITHME DE QUANTIFICATION VECTORIELLE SPHERIQUE A PARTIR DU RESEAU DE GOSSET D'ORDRE 8 SPHERICAL VECTOR QUANTIZATION ALGORITHM BASED ON AN EIGHT DIMENSIONAL GOSSET LATTICE" ANNALES DES TELECOMMUNICATIONS, LES MOULINEAUX, FR, vol. 43, no. 3/4, 1 mars 1988 (1988-03-01), pages 172-186, XP000572495	1-31

RAPPORT DE RECHERCHE INTERNATIONALE

Renseignements relatifs aux membres de familles de brevets

[redacted]ande Internationale No
PCT/FR2004/000219

Document brevet cité au rapport de recherche	Date de publication	Membre(s) de la famille de brevet(s)	Date de publication
WO 03103151 A	11-12-2003	CA 2388358 A1	30-11-2003
		WO 03103151 A1	11-12-2003